When this report is no longer needed return it to the originator.

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

The contents of this report are not to be used for advertising, publication, or promotional purposes. Citation of trade names does not constitute an official endorsement or approval of the use of such commercial products.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 1992	Final report

4. TITLE AND SUBTITLE
Vendor Information System (VIS) Systems Manual

5. FUNDING NUMBERS

6. AUTHOR(S)
Patti S. Duett, Monique F. Harrison

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
USAE Waterways Experiment Station, Information Technology Laboratory, 3909 Halls Ferry Road, Vicksburg, MS 39180-6199

8. PERFORMING ORGANIZATION REPORT NUMBER
Technical Report ITL-92-5

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Available from National Technical Information Service, 5285 Port Royal Road, Springfield, VA 22161.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This manual provides the necessary support documentation to the personnel responsible for either maintenance or support of the Vendor Information System (VIS). The manual consists of documentation for the menu system and individual programs plus a complete data base dictionary.

14. SUBJECT TERMS			15. NUMBER OF PAGES 160
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED		

PREFACE

This manual provides the necessary support documentation to the personnel responsible for either maintenance or support of the Vendor Information System (VIS). The manual consists of documentation for the menu system and individual programs plus a complete data base dictionary.

The information for this manual was compiled at the US Army Engineer Waterways Experiment Station (WES) by Patti S. Duett and Monique F. Harrison, Systems Modernization Unit (SMU), Computer Science Division (CSD), Information Technology Laboratory (ITL). Mrs. Barbara Comes was Chief of SMU, Dr. Windell Ingram was Chief of CSD, and Dr. N. Radhakrishnan was Director of ITL. Mr. Elvin E. McFerrin was Chief of the Finance and Accounting Branch, and Mr. Wayne J. Sutter was Chief of the Resource Management Office. The special efforts of Dr. Jerome Mahloch, Program Manager, ITL, who was of great assistance in the preparation of this manual, are gratefully acknowledged.

At the time of publication of this report, Director of WES was Dr. Robert W. Whalin. Commander and Deputy Director was COL Leonard G. Hassell, EN.

TABLE OF CONTENTS

1. INTRODUCTION 7
2. VIS MENU SYSTEM 9
    2.1 Background. 9
    2.2 Program Execution 10
    2.3 Role Maintenance 10
3. SQL*Forms 11
    3.1 V1_INP 11
    3.2 V3_INP 17
    3.3 V4_INP 21
    3.4 V5_INP 22
    3.5 V8_INP 24
    3.6 V9_INP 29
    3.7 V10_INP 34
    3.8 V11_INP 37
    3.9 V12_INP 40
    3.10 V14_INP 41
    3.11 V15_INP 42
    3.12 V16_INP 43
    3.13 V17_INP 44
    3.14 V18_INP 45
    3.15 V19_INP 46
    3.16 V20_INP 47
    3.17 V24_INP 48
    3.18 V25_INP 49
    3.19 V26_INP 50

3.20 V30_INP 51
3.21 V32_INP 52
3.22 V33_INP 53
3.23 V34_INP 55
3.24 V37_INP 56
3.25 V39_INP 57
3.26 V40_INP 58
3.27 V41_INP 59
3.28 V43_INP 60
3.29 V46_INP 62
3.30 V47_INP 63
3.31 V48_INP 69
3.32 MAET_INP 71
3.33 MAPP_INP 72
3.34 MCOO_INP 73
3.35 MEMP_INP 74
3.36 MOCC_INP 75
3.37 MORG_INP 76
3.38 MVEND_INP 77
3.39 MVIS_INP 78
4. SQL*Plus 80
4.1 CHTRV_SQL 80
4.2 COEMIS_SQL 81
4.3 DEOBLG_SQL 82
4.4 DEOBLIGATE_SQL 83
4.5 DISBURSEMENT_SQL 84
4.6 EXPFILE 85
4.7 FIXPAYCOLL_SQL 86
4.8 PRE1166_SQL 87

4.9 PRE1166TRV_SQL 88
4.10 PREV23RPT_SQL 89
4.11 TBO_SQL 90
4.12 UPOBLG_SQL 91
4.13 UPVEND_SQL 92
4.14 V27_SQL 93
5. SQL*Loader 95
5.1 V22_CTL 95
5.2 V28A_CTL 96
5.3 V28B_CTL 97
5.4 V44_CTL 98
5.5 VISCIV_CTL 99
5.6 VISDF_CTL 100
6. SCL PROCEDURES 101
6.1 BATCH_TRV 101
6.2 CC_BAT 102
6.3 DEOBLG_BAT 103
6.4 EXP_BAT 104
6.5 PAID_MASTER_BAT 105
6.6 PARTPAY_BAT 106
6.7 TRAVEL_BAT 107
7. SQL*ReportWriter 108
7.1 PARTPAY_REP 108
8. PRO*C 110
8.1 V23_PC 110
8.2 V31_PC 116
8.3 V42_PC 120
8.4 V45_PC 126

APPENDIX A - TABLE DEFINITIONS A-1
APPENDIX B - DATA DICTIONARY B-1

1. INTRODUCTION

This manual is intended to provide the necessary support documentation to the personnel responsible for either maintenance or support of the Vendor Information System (VIS). It is not intended to be either a comprehensive design manual or a tutorial for the programming languages comprising the suite of programs known as VIS. However, this manual can be used in conjunction with the appropriate program source listings to yield an understanding of how VIS works. The manual consists of documentation for the menu system and individual programs plus a complete data base dictionary.

While the menu system is part of VIS in every sense of the word, its purpose is significantly different from the remaining programs in VIS; consequently, a slightly different format is used to describe the program. The data base dictionary provides a listing of all data base tables, their attributes, and definition for all attributes. This will allow the reader to readily ascertain the principle relationships in VIS.

The bulk of this manual is dedicated to specific documentation of individual pr(grams comprising VIS. This documentation includes the program name, language, a brief statement of purpose, and the entry point. Also included is a description of input data requirements and output generated by the program unless I/O is satisfied by interactions with the database. A brief description of error handling procedures is also presented. In general, error handling for programs written in SQL*Forms is straightforward, but for other programs, particularly "batch" programs, care is taken to define the logical unit of work so that the user has a concept of what happens in terms of the data base upon abnormal program termination.

The most critical part of individual program documentation is that section describing program execution. This section is generally written as a "pseudo-code" or algorithmic description of transaction processing within the program. Another important aspect included in this section are

the integrity checks performed by each program. These two aspects of VIS program documentation are critical for support of maintenance activities.

The last section in the individual program documentation is called 'Other Program Notes'. This section includes important aspects of each program that do not fit in previous sections or that deserve the individual emphasis because of their importance. This area includes locking procedures for the database, use of constants in programs, hints on optimization for performance, and other operating system/RDBMS specific requirements for successful program execution.

2. VIS MENU SYSTEM

2.1 Background

The menuing system for VIS handles both role security and menu navigation for VIS. This functionality is somewhat distinct from other programs in VIS and we will depart from our conventional format to more easily explain how it works.

Security is approached in VIS by limiting access to the data base by users (other than the DBA) thru a menu system. Furthermore, data base access is restricted to specific user roles that in turn have selected access to programs to execute their specific functionality. This level of security is by no means exhaustive but does provide a degree of protection against unauthorized access to the VIS data base. The menu system and associated maintenance consists of 5 programs written in either SQL*Forms or PRO*C. Initial entry point upon user logon to VIS is through VMREC. A summary of each program is presented below:

Program Name	Program Language	Program Purpose
VMREC	SQL*Forms	Role Selection
MRECA	PRO*C	Database Connect
VMMAIN	SQL*Forms	Menu Navigation
VMRI	SQL*FORMS	Role Maintenance
MRIA	PRO*C	Grant Issuance

The menu system is table based and maintenance of the menu tables must be accomplished by the System Administrator using SQL*Plus. All program output is directed to the user's monitor and there are no specific input requirements beyond those requested by the program. Program termination occurs for all instances of an ORACLE error.

2.2 Program Execution

(1) User logs on to VIS using his/her assigned userid and password.

(2) Userid is used as key to select authorized roles from VUSERID_ROLE. If no valid roles exist, an error message is displayed and the program is terminated.

(3) Else, valid roles are displayed on screen generated by VMREC and the user is prompted to make a selection. From this selection the connect string is built and a call is made to MRECA.

(4) MRECA issues a disconnect from the database for the current user and issues a reconnect for the role selected in step (3) using an encrypted password. Control is passed back to VMREC, which calls VMMAIN.

(5) VMMAIN is used for menu navigation specific to the role currently accessing the database. Menu selections are presented in a hierarchical fashion ordered by a menu order assigned to each program accessed through the menu system. VMMAIN also maintains the original userid used in step (1) in a global area for an audit trail. Upon a selection VMMAIN calls the selected program and transfers control to it.

2.3 Role Maintenance

(1) Maintenance of the table containing the valid roles and encrypted passwords is done in VMRI. All updates to this table are marked by a time stamp. Prior to commit, the grant string for the affected role is generated and control is passed to MRIA.

(2) MRIA receives the grant string created in step B(1) and immediately executes the SQL command. Control is then returned to VMRI.

(3) VMRI enforces the unique constraint for keycode and role during any maintenance activity.

3. SQL*Forms

3.1 V1_INP

A. **Program Name**: V1_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Enter detail obligation information and create a transaction for COEMIS update.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

(1) Input Values:

(a) Commitment Document Number (Page 1)

(b) Obligation Document Number (Page 1) - Validate against OBLIGATION table. If it exists display a message stating obligation already exists.

(c) Contract Number (Page 1)

(d) Contract Modification Number (Page 1) - Validate against CONTRACT table. If it exists display a message stating that modification number already exists. If the modification number is valid go to page 2 and enter contract detail information. If it is not a true contract obligation then leave the field blank and continue to enter information on page 1.

(e) Vendor Code (Page 1) - Validate against VENDOR table. If invalid display a message stating vendor code is invalid. Provide a list values key todisplay a list of valid vendors.

(f) Vendor Address Code (Page 1) - Validate against VENDOR_ADDR table. If invalid display a message stating vendor address code is invalid. Provide a list values key to display a list of valid vendor address codes.

(g) Organization Code (Page 1) - Validate against ORGANIZATION table. If invalid display a message stating organization code is invalid. Provide a list values key to display valid organization codes.

(h) Obligation Date (Page 1) - Default to the system date. Edit to make sure the obligation date is less than or equal to the system date. Display a message stating the obligation date must be before today's date but not later.

(i) ADP Work Code (Page 1) - The following determines the type of funding: Position 1 = A,B,C,D,E,F,G,H or I -Civil; Position 1 = J,K,L,M,N,O,P,Q,R,S or T - Military; Position 1 and 2 = VW - Revolving; Position 1, 2 and 3 = VW7 - Prip. If it is Military validate against the D_F_FILE table. If not valid, display a message stating ADP Work Code is not in D-F File. If it is valid select the loc_appn_nbr into the local appropriation number from the D_F_FILE. If it is Civil or Revolving Fund, validate against the CIVRF table. If not valid, display a message stating invalid ADP work code. If it is valid select the loc_appn_nbr into the local appropriation number from the CIVRF table.

(j) Local Appropriation Number (Page 1) - Validate against APPROPRIATION table. Display a message stating invalid appropriation.

(k) Amount (Page 1) - Amount must be between 0.00 and 9999999999.99. Default to contract amount. If a contact mod number has been entered validate that the amount entered is less than or equal to the contract amount. Display a message stating you cannot obligate more than your contract amount.

(l) Item Code (Page 1) - Validate against ACCT_ELM_TYPE table. Display message stating invalid item code. If Civil or Revolving, item code must be between 202 and 399. If Military, item code must be 500 or between 202 and 399. If Prip, item code must be 474, 477, 479, or 498. Provide a list values key to display valid item codes.

(m) Object Class Code (Page 1) - Validate against the OBJECT_CLASS_CODES table. Display a message stating object class code is invalid. Provide a list values key to display a list of valid object class codes.

(n) Contract Reference Modification Number (Page 2) - Default to contract mod number on page 1.

(o) Amount (Page 2) - Must be between 0.00 and 9999999999.99

(p) Date of Award (Page 2) - Default to system date. Validate that date of award is less than or equal to the ending date. Display message stating date of award must be before ending date.

(q) Ending Date (Page 2) - Default to system date. Validate that ending date is greater than or equal to the award date. Display message stating ending date must be after date of award.

(r) Retained Percent (Page 2) - Must be between 0.00 and 99.0

(s) Discount Terms Code (Page 2) - Validate that first position must not be zero and display message stating this. Provide list values key to display list of valid discount terms.

(t) Tax ID (Page 2)

(u) Contracting Officer Representative (Page 2)

(2) Display Values:

(a) Vendor Name (Page 1) - Retrieve from VENDOR table based on vendor code.

(b) Vendor Phone (Page 1) - Retrieve from VENDOR_ADDR table based on vendor address code.

(c) Vendor Address (Page 1) - Retrieve from VENDOR_ADDR table based on vendor address code.

(d) Vendor State (Page 1) - Retrieve from VENDOR_ADDR table based on vendor address code.

(e) Vendor Zip (Page 1) - Retrieve from VENDOR_ADDR table based on vendor address code.

(f) Vendor Country (Page 1) - Retrieve from VENDOR_ADDR table based on vendor address code.

(3) Inserts:

(a) OBLIGATION table - Validate positions 2 & 3 of the obligation number against the CLASS_OF_OBLIGATION table. Display a message stating invalid class of obligation.

amount = :obligation.amount

entry_date = :obligation.entry_date

init_entry_id = user logged on

object_class_code = :obligation.object_class_code

oblg_doc_nbr = :obligation.oblg_doc_nbr

vendor_addr_code = :obligation.vendor_addr_code

vendor_code = :obligation.vendor_code

net_unliq_bal = :obligation.amount

adp_work_code = :obligation.adp_work_code

comt_doc_nbr = :control.comt_doc_nbr

org_code = :obligation.org_code

civ_mil_rvolvg_flag = :obligation.civ_mil_rvolvg_flag

status = 'C'

loc_appn_nbr = :obligation.loc_appn_nbr

contr_mod_nbr = :control.contr_mod_nbr

contr_nbr = :control.contr_nbr

(b) CONTRACT table

amount = :contract.amount

contr_mod_nbr = :control.contr_mod_nbr

contr_nbr = :control.contr_nbr

contr_ref_mod_nbr = :contract.contr_ref_mod_nbr

entry_userid = user logged on

vendor_code = :obligation.vendor_code

date_of_award = :contract.date_of_award

disc_terms_code = :contract.disc_terms_code

ending_date = :contract.ending_date

fob_date = :contract.fob_date

ret_perc = :contract.ret_perc

ret_perc_doc_nbr = :contract.ret_perc_doc_nbr

tax_id = :contract.tax_id

contr_officer_rep = :contract.contr_officer_rep

(c) CMR_4480 table - DO NOT INSERT for Revolving Fund.

acpt_date = entry_date

adp_work_code = For Civil :obligation.adp_work_code. For Military first 10 digits of :obligation.adp_work_code.

amount = :obligation.amount

batch_nbr = Civil = 37, Military = 42.

civ_mil_rvolvg_flag = :obligation.civ_mil_rvolvg_flag

dest = 'A'

dist_code = '4'

doc_nbr = :obligation.oblg_doc_nbr

interface_acpt_flag = 'I'

record_code = '02'

trns_code = 'JA'

trns_type_indic = 'O'

item_code = Null

object_class_code = :obligation.object_class_code

org_code = For civil :obligation.org_code, military '00'

other_ref_nbr = null

pay_coll_code = For civil null. For Military if 2nd and 3rd positions of obligation number are 47, 52, 92, 87 or 90 then '1' else '3'.

ref_doc_nbr = :obligation.comt_doc_nbr

trf_date = null

trns_month = null

3.2 V3_INP

A. **Program Name**: V3_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Edit or adjust detail obligation information and create a transaction for COEMIS update.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

(1) Input Values:

(a) Obligation Document Number - Validate against OBLIGATION table. Display message stating obligation does not exist. Provide a list values key to display a list of valid obligation numbers. (User must enter the obligation number or contract and contract mod number.)

(b) Contract Number - Validate against OBLIGATION table. Display a message stating that the contract number is invalid. Provide a list values key to display valid contract numbers and contract modification numbers. (User must enter the obligation number or contract and contract mod number.)

(c) Contract Mod Number

(d) Vendor Code - Retrieved from OBLIGATION table based on obligation number or contract and contract mod number. May be updated. Validate against VENDOR table and display message stating vendor is invalid. Provide list values key displaying list of valid vendors.

(e) Vendor Address Code - Retrieved from VENDOR table based on vendor code. May be updated. Validate against VENDOR_ADDR table and display message stating vendor address code is invalid. Provide a list values key to display valid address codes.

(f) Object Class Code - Retrieved from OBLIGATION table based on obligation

number or contract and contract mod number. May be edited. Validate against OBJECT_CLASS_CODES table. Display message stating that object class code is invalid. Provide a list values key to display a list of valid object class codes.

(g) Item Code - Retrieved from OBLIGATION table based on obligation number or contract and contract mod number. May be edited. Validate against ACCT_ELM_TYPE table. Display message stating invalid item code. Provide list values key to display all valid item codes.

(h) Commitment Document Number - Retrieved from OBLIGATION table based on obligation number or contract and contract mod number.

(i) Adjustment Amount - Amount obligation is to be adjusted. Validate that this amount is not more than the obligation unliquidated balance and not exceeding the contract amount.

(2) Display Values:

(a) Entry Date - Retrieved from OBLIGATION table based on obligation number or contract and contract mod number.

(b) Vendor Name - Retrieved from VENDOR table based on vendor code.

(c) Vendor Address - Retrieved from VENDOR_ADDR table based on vendor address code.

(d) Vendor City - Retrieved from VENDOR_ADDR table based on vendor address code.

(e) Vendor State - Retrieved from VENDOR_ADDR table based on vendor address code.

(f) Vendor Zip - Retrieved from VENDOR_ADDR table based on vendor address code.

(g) Vendor Country - Retrieved from VENDOR_ADDR table based on vendor address code.

(h) ADP Work Code - Retrieved from OBLIGATION table based on obligation number or contract and contract mod number.

(i) Net Unliquidated Balance - Retrieved from OBLIGATION table based on obligation number or contract and contract mod number.

(j) Gross Amount - Amount retrieved from OBLIGATION table based on obligation number or contract and contract mod number.

(3) Updates:

(a) OBLIGATION table

Amount = Amount + Adjustment Amount

Net Unliquidated Balance =

Net Unliquidated Balance + Adjustment Amount

(b) CONTRACT table - If record exists in contract table for contract number and contract mod number then vendor code = :obligation.vendor_code.

(c) CMR_4480 table - ONLY UPDATE for Civil and Military. ONLY UPDATE if adjustment amount is greater than zero.

Amount = Amount + Adjustment Amount where doc_nbr = :obligation.oblg_doc_nbr and interface_acpt_flag = 'I'.

(4) Inserts:

(a) CMR_4480 table - DO NOT insert for Revolving Fund.

Insert if acpt_flag = 'A'

acpt_date = sysdate

adp_work_code = For Civil :obligation.adp_work_code. For Military first 10 digits of :obligation.adp_work_code.

amount = :obligation.adjustment_amount

batch_nbr = Civil & Negative = 40, Civil & Positive = 37, Military & Negative = 45, Military & Positive = 42

civ_mil_rvolvg_flag = :obligation.civ_mil_rvolvg_flag

dest = 'A'

dist_code = '4'

doc_nbr = :obligation.oblg_doc_nbr

interface_acpt_flag = 'I'

record_code = '02'

trns_code = 'JA'

trns_type_indic = 'O'

item_code = null

object_class_code = :obligation.object_class_code

org_code = For Civil :obligation.org_code, Military '00'

other_ref_nbr = null

pay_coll_code = For Civil null. For Military if 2nd and 3rd positions of obligation number are 47, 53, 92, 87 or 90 then '1' else '3'.

ref_doc_nbr = :obligation.comt_doc_nbr

trf_date = null

trns_month = null

3.3 V4_INP

A. **Program Name**: V4_INP

B. **Program Language**: SQL*Forms

C. **PURPOSE**: Display obligations by ADP work code.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

(1) Input Values:

(a) ADP work code - Validate against OBLIGATION table. Display message stating ADP work code is invalid.

(2) Display Values:

(a) Total Unliquidated Obligation - Sum net_unliq_bal from OBLIGATION table based on the ADP work code entered.

(b) Obligation Number - Retrieve from OBLIGATION table based on ADP work code entered. Provide a key to call screen V5 - Display Obligations and Expenditures.

(c) Contract Number - Retrieve from OBLIGATION table based on ADP work code entered.

(d) Contract Mod Number - Retrieve from OBLIGATION table based on ADP work code entered.

(e) Commitment Number - Retrieve from OBLIGATION table based on ADP work code entered.

(f) Gross Obligation Amount - Retrieve amount from OBLIGATION table based on ADP work code entered.

(g) Net Unliquidated Balance - Retrieve from OBLIGATION table based on ADP work code entered.

I. **Other Program Notes**: Order by entry date.

3.4 V5_INP

A. **Program Name**: V5_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Display obligations and expenditures.

D. **Entry Point**: Executed through VIS menu and called from V4 (Display Obligations by ADP Work Code).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

(1) Input Values:

(a) Obligation Document Number - Validate against OBLIGATION table. Display message stating obligation does not exist. Provide a list values key to display a valid list of obligation numbers.

(2) Display Values:

(a) ADP Work Code - Retrieve from OBLIGATION table based on obligation number entered.

(b) Gross Obligation Amount - Retrieve amount from OBLIGATION table based on obligation number entered.

(c) Commitment Number - Retrieve from OBLIGATION table based on obligation number entered.

(d) Contract Number - Retrieve from OBLIGATION table based on obligation number entered.

(e) Contract Mod Number - Retrieve from OBLIGATION table based on obligation number entered.

(f) ADP Work Code - Retrieve from OBLIGATION table based on obligation number entered.

(g) Net Unliquidated Balance - Retrieve from OBLIGATION table based on obligation number entered.

(h) Vendor Code - Retrieve from OBLIGATION table based on obligation number entered.

(i) Vendor Name - Retrieve from VENDOR table based on vendor code.

(j) Initiator - Retrieve from EMPLOYEE table based on init_entry_id in the OBLIGATION table for the obligation number entered.

(k) Expenditure Number - Retrieve from EXPENDITURE table based on the obligation number entered. Provide a key to call screen V39 - Display expenditure information.

(l) Entry Date - Retrieve from EXPENDITURE table based on the obligation number entered.

(m) Expenditure Amount - Retrieve from EXPENDITURE table based on the obligation number entered.

(n) Scheduled Payment Date - Retrieve from EXPENDITURE table based on the obligation number entered.

(o) Disbursement - Sum amount from DISBURSEMENT table for each expenditure document number. If the expenditure has not been disbursed default the disbursement amount to 0.00.

I. **Other Program Notes**: Order expenditure information by expenditure document number.

3.5 V8_INP

A. **Program Name**: V8_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Enter detail expenditure information and create a transaction for COEMIS update.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

(1) Input Values:

(a) Obligation Document Number - Validate against OBLIGATION table. Display message stating obligation is invalid. (User must enter the obligation number or contract and contract mod number.)

(b) Contract Number - Validate against OBLIGATION table. Display a message stating that the contract number is invalid. (User must enter the obligation number or contract and contract mod number.)

(c) Contract Mod Number

(d) Delivery Date - Default to the system date. Validate that delivery date is less than or equal to the system date. Display message. Edit to see if delivery date is less than or equal to the date of the obligation. Display a WARNING message but continue processing.

(e) Acceptance Date - Default to the system date. Validate that the acceptance date is equal to or greater than the delivery date. Display message.

(f) Description - Allow user to enter two lines of descriptions.

(g) Partial Final Flag - Only valid entries are P and F. Default to P. Check to see if the partial final flag is set to F in the EXPENDITURE table for the obligation entered. Display message stating final expenditure exists for this obligation and no new expenditure is allowed.

(h) TBO Flag - Only valid entries are Y and N. Default to N. Validate that Y

is entered only for Military funding. Display message stating the source of funds must be Military.

(i) DSSN Number - This field must be entered if the TBO flag is Y otherwise do not allow this field to be entered.

(j) Cycle Number - This field must be entered if the TBO flag is Y otherwise do not allow this field to be entered.

(k) Amount - The amount must not be greater than the obligation net unliquidated balance. Display a message stating that this amount cannot exceed the net unliquidated balance.

(2) Display values:

(a) Net Unliquidated Balance - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(b) Object Class Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(c) Item Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(d) ADP Work Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(e) Vendor Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(f) Vendor Name - Retrieved from VENDOR table based on vendor code.

(g) Organization Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(h) Partial Pay Number - Retrieve max partial pay number from EXPENDITURE tabled based on the obligation number and add 1 to it.

(3) Inserts:

(a) EXPENDITURE table

amount = :expenditure.amount

entry_date = sysdate

exp_doc_nbr = generate expenditure document number.

1st position = F

2nd and 3rd positions = Fiscal month

4th and 5th positions = Fiscal year

6th - 9th positions = Generated sequence number (The sequence number is generated by selecting the nextval from the EXPSEQ sequence)

init_entry_id = user logged on

oblg_doc_nbr = :obligation.oblg_doc_nbr

partial_final_flag = :expenditure.partial_final_flag

vendor_addr_code = :obligation.vendor_addr_code

vendor_code = :obligation.vendor_code

adp_work_code = :obligation.adp_work_code

status = 'C'

cycle_nbr = :expenditure.cycle_nbr

acpt_date = :expenditure.acpt_date

del_date = :expenditure.del_date

descr1 = :expenditure.descr1

descr2 = :expenditure.descr2

disc_terms_code = null

dssn_nbr = :expenditure.dssn_nbr

inv_nbr = null

partial_pay_nbr = :expenditure.partial_pay_nbr

scheduled_paymnt_date = null

tbo_flag = :expenditure.tbo_flag

(b) CMR_4480 table - For Expenditure

acpt_date = entry_date

adp_work_code = :obligation.adp_work_code

batch_nbr = Civil = 55, Military = 60, Revolving = 47

civ_mil_rvolvg_flag = :obligation.civ_mil_rvolvg_flag

dest = 'A'

dist_code = '4'

doc_nbr = generated :expenditure.exp_doc_nbr

interface_acpt_flag = 'I'

record_code = '02'

trns_code = Civil & Military - 'MA', Revolving - 'MD'

trns_type_indic = 'E'

item_code = :obligation.item_code

object_class_code = :obligation.object_class_code

org_code = :obligation.org_code

other_ref_nbr = null

pay_coll_code = For Civil null. For Military if 2nd and 3rd positions of the obligation number are 47, 53, 92, 87, or 90 then '1' else '3'.

ref_doc_nbr = For Civil & Military = :Obligation.oblg_Doc_nbr. Revolving = null

trf_date = null

trns_month = null

(c) CMR_4480 table - Only insert if TBO flag = 'Y'.

acpt_date = entry_date

adp_work_code = Retrieve adp_wc_d from D_F_FILE where adp_wc_g = :Obligation.adp_work_code

amount = :expenditure.amount

batch_nbr = '86'

dist_code = '4'

doc_nbr = Generate document number

1st position = K

2nd and 3rd positions = Fiscal month

4th - 9th positions = 022079

object_class_code = :Obligation.object_class_code

org_code = '00'

record_code = '02'

ref_doc_nbr = :expenditure.exp_doc_nbr

interface_acpt_flag = 'I'

trns_code = 'PF'

civ_mil_rvolvg_flag = 'M'

trns_type_indic = 'B'

dest = 'A'

other_ref_nbr = generate reference document number

1st and 2nd positions = :expenditure.cycle_nbr

3rd - 5th positions = 090

6th - 9th positions = :expenditure.dssn_nbr

pay_coll_code = '1'

(d) DISBURSEMENT table - Only insert if TBO flag = 'Y'

amount = :expenditure.amount

disb_date = sysdate

disb_doc_nbr = generate disbursement number

1st position = K

2nd and 3rd positions = Fiscal month

4th - 9th positions = 022070

exp_doc_nbr = :expenditure.exp_doc_nbr

(4) Updates:

(a) OBLIGATION table

Net_unliq_bal = Net_unliq_bal - :Expenditure.amount

3.6 V9_INP

A. **Program Name**: V9_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Schedule expenditures for payment and create necessary transaction for COEMIS update.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

(1) Input Values:

(a) Expenditure Document Number - Validate against EXPENDITURE table. Display message stating expenditure is invalid. Provide a list values key to display a list of valid expenditure document numbers. Provide a key to call V33 - Enter Invoice. Check to make sure expenditure has not been previously scheduled. Display message stating expenditure has already been scheduled.

(b) Description - Allow user to enter two lines of descriptions.

(c) Vendor Address Code - Validate against VENDOR_ADDR table. Display message stating vendor address code is invalid. Provide a list values key to display a list of valid vendor address codes.

(d) Partial Final Flag - Valid choices are P or F. Display message if invalid.

(e) Invoice Number - Calls screen V34 - Invoice lookup. Copy invoice number from screen V33 which was called from the expenditure document number.

(f) Vendor Tax ID

(g) Disbursement Amount

(h) Scheduled Payment Date

(i) Delivery Date

(j) Acceptance Date

(2) Display Values :

(a) ADP Work Code - Retrieve from EXPENDITURE table based on expenditure number entered.

(b) Vendor Code - Retrieve from EXPENDITURE table based on expenditure number entered.

(c) Vendor Name - Retrieve from VENDOR table based on vendor code.

(d) Vendor City - Retrieve from VENDOR_ADDR table based on vendor address code.

(e) Vendor State - Retrieve from VENDOR_ADDR table based on vendor address code.

(f) Vendor Zip - Retrieve from VENDOR_ADDR table based on vendor address code.

(g) Vendor Country - Retrieve from VENDOR_ADDR table based on vendor address code.

(h) Vendor Phone - Retrieve from VENDOR_ADDR table based on vendor address code.

(i) Partial Pay Number - Retrieve from EXPENDITURE table based expenditure number entered.

(3) Inserts: Validate the :Obligation.net_unliq_bal >= (:Expenditure.disb_amt - :Expenditure.amount). Display message stating the obligation unliquidated balance may not be exceeded.

(a) SCHEDULED_PAYMNT table

adp_work_code = Civil = 1st - 5th positions of
:expenditure.adp_work_code, 6th - 15th positions (zero filled)
Military = Retrieve adp_wc_d from D_F_FILE where
adp_wc_g = :expenditure.adp_work_code
Revolving = 1st and 2nd positions 'VW',
3rd - 15th positions (zero filled)

amount = :expenditure.disb_amt

exp_doc_nbr = :expenditure.exp_doc_nbr

init_entry_id = user logged on

loc_appn_nbr = :obligation.loc_appn_nbr

object_class_code = :obligation.object_class_code

scheduled_paymnt_date = :expenditure.scheduled_paymnt_date

status = 'C'

vendor_addr_code = :expenditure.vendor_addr_code

vendor_code = :expenditure.vendor_code

contr_mod_nbr = :obligation.contr_mod_nbr

contr_nbr = :obligation.contr_nbr

trns_date = entry_date

trnsf_date = null

(b) CMR_4480 table - DO NOT INSERT for Revolving Fund.

Only insert if interface_acpt_flag in CMR_4480 table is 'A' for the expenditure number entered.

acpt_date = sysdate

adp_work_code = :expenditure.adp_work_code

amount - :expenditure.disb_amt - :expenditure.amount

batch_nbr = Civil & Positive = 55

Military & Positive = 60

Civil & Negative = 56

Military & Negative = 61

civ_mil_rvolvg_flag = :obligation.adp_work_code

dest = 'A'

dist_code = '4'

doc_nbr = :expenditure.exp_doc_nbr

interface_acpt_flag = 'I'

record_code = '02'

trns_code = 'MA'

trns_type_indic = 'E'

item_code = :obligation.item_code

object_class_code = :obligation.object_class_code

org_code = :obligation.org_code

other_ref_nbr = null

pay_coll_code = For Civil null. For Military if 2nd and 3rd positions of obligation number are 47, 52, 92, 87 or 90 then '1' else '3'.

ref_doc_nbr = :obligation.oblg_doc_nbr

trf_date = null

trans_month = null

(4) Updates: Validate the :Obligation.net_unliq_bal >= (:Expenditure.disb_amt - :Expenditure.amount). Display a message stating the obligation net unliquidated balance may not be exceeded.

(a) OBLIGATION table

net_unliq_bal = net_unliq_bal - (:expenditure.disb_amt - :expenditure.amount)

(b) EXPENDITURE table

descr1 = :expenditure.descr1

descr2 = :expenditure.descr2

vendor_addr_code = :expenditure.vendor_addr_code

partial_final_flag = :expenditure.partial_final_flag

inv_nbr = :expenditure.inv_nbr

scheduled_paymnt_date = :expenditure.scheduled_paymnt_date

del_date = :expenditure.del_date

acpt_date = :expenditure.acpt_date

(c) CMR_4480 table - DO NOT UPDATE for Revolving Fund.

Only update if interface_acpt_flag is 'I' in CMR_4480 table for the expenditure number entered.

amount = :expenditure.disb_amt

(d) VENDOR table

vendor_tax_id = :expenditure.vendor_tax_id

3.7 V10_INP

A. **Program Name**: V10_inp

B. **Program Language**: SQL*Forms

C. **Purpose**: Cancel an expenditure and create transaction for COEMIS update.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

(1) Input Values:

(a) Expenditure Document Number - Validate against EXPENDITURE table. Display message stating expenditure number is invalid. Validate against EXPENDITURE table to insure expenditure has not previously been canceled. Display message stating expenditure has already been canceled. Provide list of values key to display a valid list of expenditure numbers to be canceled.

(b) Cancel Expenditure Flag - Ask the question do you want to cancel this expenditure. Valid entries are Y or N. Only cancel if Y is entered.

(c) Description - Allow user to enter one line of description.

(2) Display Values:

(a) Vendor Name - Retrieve from VENDOR table based on vendor code from EXPENDITURE table.

(b) ADP Work Code - Retrieve from EXPENDITURE table based on expenditure number entered.

(c) Amount - Retrieve from EXPENDITURE table based on expenditure number entered.

(d) New Expenditure Number - Generate new expenditure number for revolving fund if expenditure entered has already gone to COEMIS.

(3) Inserts:

(a) CMR_4480 table - ONLY INSERT if the interface_acpt_flag is 'A' for the

expenditure number entered. If Revolving, then a new expenditure document number must be generated.

acpt_date = sysdate

adp_work_code = Civil & Military = :expenditure.adp_work_code,
Revolving = 'VW0000000000000'

amount = (-:Expenditure.amount)

batch_nbr = Civil = '52', Military = '57', Revolving = '50'

civ_mil_rvolvg_flag = :expenditure.civ_mil_rvolvg_flag

dest = 'A'

dist_code = '4'

doc_nbr = Civil & Military = :expenditure.exp_doc_nbr, Revolving = :expenditure.new_exp_doc_nbr

interface_acpt_flag = 'I'

record_code = '02'

trns_code = Civil & Military = 'MA', Revolving = 'MC'

trns_type_indic = 'E'

item_code = :obligation.item_code

object_class_code = :obligation.object_class_code

org_code = :obligation.org_code

pay_coll_code = For Civil null. For Military & Revolving if 2nd and 3rd positions of obligation number are 47, 52, 92, 87 or 90 then '1' else '3'.

ref_doc_nbr = Civil & Military = :obligation.oblg_doc_nbr, Revolving = :expenditure.exp_doc_nbr

trf_date = null

trns_month = null

(4) Updates:

(a) OBLIGATION table

net_unliq_bal = net_unliq_bal + :expenditure.amount

(b) CMR_4480 table - ONLY UPDATE if the interface_acpt_flag = 'I' for the expenditure number entered.

(c) EXPENDITURE table

amount = '0.00'

status = 'X'

3.8 V11_INP

A. **Program Name**: V11_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Establish and edit recurring expenditure.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

(1) Input Values:

(a) Contract Number - Validate against OBLIGATION table that the contract and contract mod number entered references a valid obligation. Display message stating that contract and mod number does not reference an obligation. (User must enter contract and contract mod number.)

(b) Contract Mod Number - (User must enter contract and contract mod number.)

(c) Description - Allow user to enter two lines of descriptions.

(d) Start Date - Edit to make sure the start date is greater than or equal to the system date. Display message.

(e) Expiration Date - Edit to make sure the expiration date is greater than or equal to the start date. Display message.

(f) Number of Payments

(g) Payment Period

(h) Incremental Expenditure Amount - Validate that the amount is less than or equal to the net unliquidated obligation balance.

(i) Status - Valid entries are A for active or I for inactive. Display message.

(2) Display Values:

(a) Obligation number - Retrieved from OBLIGATION table based on contract and contract mod number entered.

(b) Net Unliquidated Balance - Retrieved from OBLIGATION table based on

contract and contract mod number entered.

(c) Object Class Code - Retrieved from OBLIGATION table based on contract and contract mod number entered.

(d) Item Code - Retrieved from OBLIGATION table based on contract and contract mod number entered.

(e) Vendor Code - Retrieved from OBLIGATION table based on contract and contract mod number entered.

(f) Vendor Name - Retrieved from VENDOR table based on vendor code.

(g) Vendor Address Code - Retrieved from OBLIGATION table based on contract and contract mod number entered.

(h) ADP Work Code - Retrieved from OBLIGATION table based on contract and contract mod number entered.

(i) Vendor Address - Retrieved from VENDOR_ADDR table based on vendor address code.

(j) Vendor City - Retrieved from VENDOR_ADDR table based on vendor address code.

(k) Vendor State - Retrieved from VENDOR_ADDR table based on vendor address code.

(l) Vendor Zip - Retrieved from VENDOR_ADDR table based on vendor address code.

(m) Vendor Country - Retrieved from VENDOR_ADDR table based on vendor address code.

(n) Vendor Phone - Retrieved from VENDOR_ADDR table based on vendor address code.

(3) Inserts:

(a) RECURRING_EXP table

amount = :recurring_exp.amount

contr_mod_nbr = :obligation.contr_mod_nbr

contr_nbr = :obligation.contr_nbr

entry_date = sysdate

expr_date = :recurring_exp.expr_date

init_entry_id = user logged on

item_code = :obligation.item_code

no_paymnts = :recurring_exp.no_paymnts

object_class_code = :obligation.object_class_code

oblg_doc_nbr = :obligation.oblg_doc_nbr

paymnt_period = :recurring_exp.paymnt_period

start_date = :recurring_exp.start_date

status = :recurring_exp.status

vendor_addr_code = :obligation.vendor_addr_code

adp_work_code = :obligation.adp_work_code

descr1 = :recurring_exp.descr1

descr2 = :recurring_exp.descr2

next_exp_date = :recurring_exp.start_date

3.9 V12_INP

A. **Program Name**: V12_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup valid object class codes and descriptions.

D. **Entry Point**: Called from V1 (Create Obligation) and V3 (Edit/Adjust Obligation).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select and display records containing the object class code and description from the OBJECT_CLASS_CODES table based on the Civil, Military, Revolving flag that is passed from the calling form. If the Civil, Military, Revolving flag = 'C' and Civil flag = 'Y' in the OBJECT_CLASS_CODES table, then display the Civil object class codes. If the Civil, Military, Revolving flag = 'M' and Military flag = 'Y' in the OBJECT_CLASS_CODES table, then display the Military object class codes. If the Civil, Military, Revolving flag = 'R' and Revolving flag = 'Y' in the OBJECT_CLASS_CODES table, then display the Revolving Fund object class codes. The records are ordered by the object class code.

Once records are retrieved and displayed, the user can pick an object class code from the list of valid codes. The object class code and description are passed to the calling form.

3.10 V14_INP

A. **Program Name**: V14_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup valid organization codes based on the organization name.

D. **Entry Point**: Called from V1 (Create Obligation) and V3 (Edit/Adjust Obligation).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter all or a portion of the organization name to be found. Select and display the organization name and organization code from the ORGANIZATION table based on the organization name entered.

Once records are retrieved and displayed, the user can pick an organization code from the list of valid codes. The organization code is passed to the calling form.

3.11 V15_INP

A. **Program Name**: V15_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup valid item codes and descriptions.

D. **Entry Point**: Called from V1 (Create Obligation), and V3 (Edit/Adjust Obligation). E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter all or a portion of the item code to be found. Select and display the item code and description from the ACCT_ELM_TYPE table based on the item code entered.

Once records are retrieved and displayed, the user can pick an item code from the list of valid codes. The item code is passed to the calling form.

3.12 V16_INP

A. **Program Name**: V16_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup valid vendor codes based on the vendor name.

D. **Entry Point**: Called from V1 (Create Obligation), V3 (Edit/Adjust Obligations), V19 (Lookup Valid Contract Numbers), V30 (Vendor Expenditure Lookup), V40 (Display Obligations by Vendor), and MVEND (Maintain Vendors and Vendor Address Information).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter all or a portion of the vendor name to be found. Select and display the vendor code and vendor name from the VENDOR table based on the vendor name entered.

Once records are retrieved and displayed, the user can pick a vendor code from the list of valid codes. The vendor code and vendor name are passed to the calling form.

3.13 V17_INP

A. **Program Name**: V17_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup valid vendor address information.

D. **Entry Point**: Called from V1 (Create Obligation), V3 (Edit/Adjust Obligations), and V9 (Scheduled Payments).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter the vendor code or the calling form passes the vendor code. Select and display the vendor code and vendor name from the VENDOR table based on the vendor code entered or passed. Also select and display the vendor address code, vendor address1, vendor address2, vendor city, vendor country, vendor state, vendor zipcode, and vendor phone from the VENDOR_ADDR table based on the vendor code entered or passed.

Once records are retrieved and displayed, the user can pick a vendor address code from the list of valid codes. If called from a form, the vendor address code is passed to the calling form.

3.14 V18_INP

A. **Program Name**: V18_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup valid obligation document numbers for a specific user.

D. **Entry Point**: Called from V3 (Edit/Adjust Obligation) and V5 (Display Obligations and Expenditures).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select and display obligation document number, initial entry date, and obligation amount from the OBLIGATION table. Select and display the initiator's name by matching the userid of the user to the userid in the EMPLOYEE table.

Once records are retrieved and displayed, the user can pick the appropriate obligation document number. This obligation document number is passed to the calling form.

3.15 V19_INP

A. **Program Name**: V19_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup valid contract numbers.

D. **Entry Point**: Called from V3 (Edit/Adjust Obligations), and V46 (Display Contract Information).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter all or a portion of the vendor code. Select and display the vendor code and vendor name from the VENDOR table based on the vendor code entered. Provide a lookup of vendors. Also select and display records containing the contract number and contract modification number from the CONTRACT table where the vendor code matches the vendor code entered. Order records by contract number, and contract modification number. Also determine whether the record is a cost or no-cost mod by checking to see of the contract and contract modification numbers exist in the OBLIGATION table. If they exist in the OBLIGATION table, display a 'Y' (Yes) otherwise display a 'N' (No).

Once records are retrieved and displayed, the user can select a contract number from the valid list. If called from a form, the contract number and contract modification number are passed to the calling form.

3.16 V20_INP

A. **Program Name**: V20_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup valid discount terms.

D. **Entry Point**: Called from V1 (Create Obligation) and V33 (Enter Invoice Information).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select and display all records containing the discount terms code, percent, and discount days from the DISC_TERMS table.

Once records are retrieved and displayed, the user can pick the appropriate discount terms. This discount terms code is passed to the calling form.

3.17 V24_INP

A. **Program Name**: V24_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Edit rejected travel expenditures.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: User can select an obligation from a list of travel obligations (V32) or enter an obligation to be edited. A record is displayed containing the obligation document number, vendor code, expenditure amount, and acceptance flag from the TRAVEL_EXP table where the acceptance flag = 'R' (Rejected), and the obligation document number in the TRAVEL_EXP table matches the obligation document number entered.

User can update the obligation document number, vendor code, expenditure amount and acceptance flag. If the vendor code is updated it is validated against the VENDOR table. If the expenditure amount is updated, the system checks to make sure that the expenditure amount is less than or equal to the net unliquidated balance of the obligation. If the acceptance flag is updated, the only valid values are 'I' (In progress) and 'R' (Rejected).

3.18 V25_INP

A. **Program Name**: V25_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Cancel disbursements.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter an expenditure document number for the disbursement to be canceled. A disbursement can be canceled only if the expenditure has not been canceled and the transfer date is null. Select and display the description, scheduled payment date, adp work code, contract number, contract modification number, vendor code, vendor name, and disbursement amount from the EXPENDITURE, SCHEDULED_PAYMNT, and VENDOR tables where the expenditure document number entered matches the expenditure document number in the EXPENDITURE and SCHEDULED_PAYMNT tables and the vendor code in the EXPENDITURE table matches the vendor code in the VENDOR table. The user is asked 'Do you want to cancel this disbursement?:'. If 'Y' (Yes), set the scheduled payment date = null, invoice number = null, status = 'C' in the EXPENDITURE table where the expenditure document number matches the expenditure document number entered.

3.19 V26_INP

A. **Program Name**: V26_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Review scheduled payment transactions by date.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter the scheduled payment date of transactions to be reviewed. Select records containing the expenditure document number, adp work code, amount, local appropriation number, contract number, contract modification number, initial entry id, object class code, vendor name, vendor address code, vendor address1, vendor address2, vendor city, vendor state, and vendor zipcode from the SCHEDULED_PAYMNT, VENDOR, and VENDOR_ADDR tables where the scheduled payment date in the SCHEDULED_PAYMNT table matches the scheduled payment date entered, the vendor code in the SCHEDULE_PAYMNT table matches the vendor code in the VENDOR table, and the vendor address code in the SCHEDULED_PAYMNT table matches the vendor address code in the VENDOR_ADDR table.

3.20 V30_INP

A. **Program Name**: V30_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup obligation and expenditure information for a specific vendor.

D. **Entry Point**: Called from V9 (Scheduled Payments).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter the vendor code. Select and display the vendor name from the VENDOR table based on the vendor code entered. Provide a lookup of vendors. Select and display records containing the expenditure document number, obligation document number, contract number, contract modification number, and amount from the EXPENDITURE and OBLIGATION tables. The expenditure document number, obligation document number and amount are retrieved from the EXPENDITURE table based on the vendor code entered and where the scheduled payment date is null. The contract number and contract modification number are retrieved from the OBLIGATION table based on the obligation document number retrieved from the EXPENDITURE table. Once records are retrieved and displayed, the user can pick an expenditure document number from the list of available documents. The expenditure document number is passed to the calling form.

3.21 V32_INP

A. **Program Name**: V32_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup rejected travel obligations.

D. **Entry Point**: Called from V24 (Edit Travel Expenditures).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select and display records containing the obligation document number, vendor code, expenditure amount, and net unliquidated balance from the TRAVEL_EXP and OBLIGATION tables. The obligation document number, vendor code and expenditure amount are selected from the TRAVEL_EXP table where the acceptance flag = 'R'. The net unliquidated balance is selected from the OBLIGATION table based on the obligation document numbers retrieved from the TRAVEL_EXP table.

Once records are retrieved and displayed, the user can pick the appropriate obligation document number. This obligation document number is passed to the calling form.

3.22 V33_INP

A. **Program Name**: V33_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Enter or edit invoice detail information.

D. **Entry Point**: Called from V9 (Scheduled Payments).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter an invoice number. The vendor code is passed from the calling form. Check to see if the invoice exists by selecting from the INVOICE table where the vendor code matches the vendor code passed and the invoice number matches the invoice number entered. If the invoice already exists, user cannot edit.

1. Input values:

   (a) Invoice Date - invoice date must be less than or equal to the system date.

   (b) Invoice Due Date - due date must be greater than or equal to the invoice date.

   (c) Invoice Receipt Date - receipt date must be greater than or equal to the invoice date and less than or equal to the system date.

   (d) Invoice Terms Code - invoice terms are validated against the DISC_TERMS table.

   (e) Gross Amount

   (f) Earnings Period Start Date - start date must be less than the system date.

   (g) Earnings Period Stop Date - stop date must be less than the system date and greater than the earnings start date.

   (h) Prompt Payment Flag - Y (Yes) or N (No) are the only valid entries.

2. Display values:

   (a) Discount Percent - selected from DISC_TERMS table based on the invoice terms code entered.

   (b) Discount Days - selected from DISC_TERMS table based on the invoice terms code entered.

3. Inserts:

Insert into the invoice table the vendor code, invoice number, invoice date, invoice due date, invoice receipt date, discount terms code, gross amount, earnings period start date, earnings period stop date, prompt payment flag, and initial entry id.

4. The invoice number is passed to the calling form.

3.23 V34_INP

A. **Program Name**: V34_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup invoice information.

D. **Entry Point**: Called from V9 (Scheduled Payments).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: The vendor code is passed from the calling form. Select and display all records containing the invoice number, vendor code, and amount from the INVOICE table based on the vendor code that was passed from the calling form.

Once records are retrieved and displayed, the user can select a particular invoice from the valid list. The invoice number and vendor code are passed back to the calling form.

3.24 V37_INP

A. **Program Name**: V37_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Lookup expenditure information

D. **Entry Point**: Called from V10 (Cancel Expenditure).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select and display records containing the expenditure document number and amount. Order records by the entry date of the expenditure.

Once records are retrieved and displayed, the user can pick an expenditure document number from the list of available documents. The expenditure document number is passed to the calling form.

3.25 V39_INP

A. **Program Name**: V39_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Display expenditure information for a specific expenditure document.

D. **Entry Point**: Executed through VIS menu. Called from V5 (Display Obligations and Expenditures).

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter the expenditure document number or the calling form passes the expenditure document number. Select and display the obligation document number, net unliquidated balance, adp work code, object class code, item code, vendor code, contract number, and contract modification number from the OBLIGATION table based on the obligation number selected from the EXPENDITURE table for the expenditure document number entered.

Select and display the vendor name from the VENDOR table based on the vendor code retrieved from the OBLIGATION table.

Select and display the description, partial/final flag, partial pay number, TBO flag, dssn number, cycle number, delivery date, acceptance date, expenditure amount, scheduled payment date, and invoice number from the EXPENDITURE table based on the expenditure document number entered.

Select and display the employee name from the EMPLOYEE table based on the initial entry id retrieved from the EXPENDITURE table.

Select and display the check number, and DOV number from the DISBURSEMENT table based on the expenditure document number entered.

3.26 V40_INP

A. **Program Name**: V40_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Display obligations for a specific vendor.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter the vendor code. Select and display the vendor name from the VENDOR table based on the vendor code entered. Provide a lookup of vendors. Select and display records containing the obligation document number, commitment document number, contract number, contract modification number, amount and net unliquidated balance from the OBLIGATION table where the vendor code entered matches the vendor code in the OBLIGATION table.

Once records are retrieved and displayed the user can select a specific obligation document and look at expenditure information through V5 (Display Obligations and Expenditures).

3.27 V41_INP

A. **Program Name**: V41_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Edit automatic scheduling transactions that have been rejected.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter an obligation to be edited. A record is displayed containing the obligation document number, vendor code, expenditure amount, and acceptance flag from the AUTO_SCHED table where the acceptance flag = 'R' (Rejected), and the obligation document number in the AUTO_SCHED table matches the obligation document number entered.

User can update the obligation document number, vendor code, expenditure amount and acceptance flag. If the vendor code is updated it is validated against the VENDOR table. The acceptance flag is set to 'I' (in progress) and the record is updated to the AUTO_SCHED table.

3.28 V43_INP

A. **Program Name**: V43_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Adjust expenditure amount for a specific expenditure document.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter an expenditure document number to be adjusted. Select and display the description, expenditure amount, and vendor code from the EXPENDITURE table based on the expenditure document number entered and where the status is not equal to 'X' (Canceled). Select and display the obligation document number, net unliquidated balance, adp work code, and vendor code from the OBLIGATION table based on the obligation document number retrieved from the EXPENDITURE table. Select and display the vendor name from the VENDOR table based on the vendor code retrieved from the OBLIGATION table. Select and display the disbursement amount from the DISBURSEMENT table based on the expenditure document number entered.

Prompt user to enter the adjustment amount. Check to make sure the amount of the adjustment + the expenditure amount is greater than or equal to 0. Also check to make sure the adjustment amount has not exceeded the net unliquidated balance for the obligation. If adjustment amount meets the above criteria, update the net unliquidated balance in the OBLIGATION table by the amount of the adjustment.

If the interface acceptance flag = 'I' (in progress) in the CMR_4480 table for the expenditure document number being adjusted, update CMR_4480 and set the amount = amount + adjustment amount.

If the interface acceptance flag <> 'I' then insert a record into the CMR_4480 table for the expenditure document number only if Civil or Military. Insert into CMR_4480 the following:

1. acceptance date = sysdate
2. adp work code = adp work code from EXPENDITURE table

3. amount = adjustment amount
4. batch number =

   Civil & Positive = 55

   Civil & Negative = 52

   Military & Positive = 60

   Military & Negative = 57
5. district code = '4'
6. document number = expenditure document number
7. item code = item code from OBLIGATION table
8. object class code = object class code from OBLIGATION table
9. organization code = organization code from OBLIGATION table
10. pay collect code =

    Civil = null

    Military or Revolving Fund =

        If 2nd and 3rd positions of the obligation document number = '47','52','87','90','92',

        the pay collect code = '3' else the pay collect code = '1'
11. record code = '02'
12. reference document number = obligation document number
13. interface acceptance flag = 'I' (in progress)
14. transaction code = 'MA'
15. civil, military, revolving flag =

    Civil = 'C', Military = 'M', Revolving Fund = 'R'
16. transaction type indicator = 'E' (Expenditure)
17. destination = 'A' (Accounting system)

3.29 V46_INP

A. **Program Name**: V46_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: View contract information.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter the contract number. Provide a lookup of valid contracts. Select records containing contract modification number, obligation document number, obligation amount, net unliquidated balance, adp work code, commitment document number, and entry date from the OBLIGATION table based on the contract number entered. Select the vendor name from the VENDOR table based on the vendor code retrieved in the OBLIGATION table.

Once records are retrieved and displayed, the user can call V5 (Display Obligations and Expenditures) for more detailed information.

3.30 V47_INP

A. **Program Name**: V47_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Enter detail retained percent document information and create a transaction for COEMIS update.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

(1) Input Values:

(a) Obligation Document Number - Validate against OBLIGATION table. Display message stating obligation is invalid. Validate against CONTRACT table that a retained percent document number does not already exist. If one exists display a message stating retained percent document number already exists. (User must enter the obligation number or contract and contract mod number.)

(b) Contract Number - Validate against OBLIGATION table. Display a message stating that the contract number is invalid. Validate against CONTRACT table that a retained percent document number does not already exist. If one exists display a message stating retained percent document number already exists. (User must enter the obligation number or contract and contract mod number.)

(c) Contract Mod Number

(d) Delivery Date - Default to the system date. Validate that delivery date is less than or equal to the system date. Display message. Edit to see if delivery date is less than or equal to the date of the obligation. Display a WARNING message but continue processing.

(e) Acceptance Date - Default to the system date. Validate that the acceptance date is equal to or greater than the delivery date. Display message.

(f) Description - Allow user to enter two lines of descriptions.

(g) Partial Final Flag - Only valid entries are P and F. Default to P. Check to see if the partial final flag is set to F in the EXPENDITURE table for the obligation entered. Display message stating final expenditure exists for this obligation and no new expenditure is allowed.

(h) TBO Flag - Only valid entries are Y and N. Default to N. Validate that Y is entered only for Military funding. Display message stating the source of funds must be Military.

(i) DSSN Number - This field must be entered if the TBO flag is Y otherwise do not allow this field to be entered.

(j) Cycle Number - This field must be entered if the TBO flag is Y otherwise do not allow this field to be entered.

(k) Amount - The amount must not be greater than the obligation net unliquidated balance. Display a message stating that this amount cannot exceed the net unliquidated balance.

(2) Display values:

(a) Net Unliquidated Balance - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(b) Object Class Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(c) Item Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(d) ADP Work Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(e) Vendor Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(f) Vendor Name - Retrieved from VENDOR table based on vendor code.

(g) Organization Code - Retrieved from OBLIGATION table based on obligation number entered or contract and contract mod number entered.

(h) Partial Pay Number - Retrieve max partial pay number from EXPENDITURE

tabled based on the obligation number and add 1 to it.

(3) Inserts:

(a) EXPENDITURE table

amount = :expenditure.amount

entry_date = sysdate

exp_doc_nbr = generate expenditure document number.

1st position = F

2nd and 3rd positions = Fiscal month

4th and 5th positions = Fiscal year

6th - 9th positions = Generated sequence number (The sequence number is generated by selecting the nextval from the EXPSEQ sequence).

init_entry_id = user logged on

oblg_doc_nbr = :obligation.oblg_doc_nbr

partial_final_flag = :expenditure.partial_final_flag

vendor_addr_code = :obligation.vendor_addr_code

vendor_code = :obligation.vendor_code

adp_work_code = :obligation.adp_work_code

status = 'C'

cycle_nbr = :expenditure.cycle_nbr

acpt_date = :expenditure.acpt_date

del_date = :expenditure.del_date

descr1 = :expenditure.descr1

descr2 = :expenditure.descr2

disc_terms_code = null

dssn_nbr = :expenditure.dssn_nbr

inv_nbr = null

partial_pay_nbr = :expenditure.partial_pay_nbr

scheduled_paymnt_date = null

tbo_flag - :expenditure.tbo_flag

(b) CMR_4480 table - For Expenditure

acpt_date = sysdate

adp_work_code = :obligation.adp_work_code

batch_nbr = Civil = 55, Military = 60, Revolving = 47

civ_mil_rvolvg_flag = :obligation.civ_mil_rvolvg_flag

dest = 'A'

dist_code = '4'

doc_nbr = generated :expenditure.exp_doc_nbr

interface_acpt_flag = 'I'

record_code = '02'

trns_code = Civil = 'MA', Military = 'ME', Revolving = 'MD'

trns_type_indic = 'E'

item_code = :obligation.item_code

object_class_code = :obligation.object_class_code

org_code = :obligation.org_code

other_ref_nbr = null

pay_coll_code = For Civil null. For Military if 2nd and 3rd positions of the obligation number are 47, 53, 92, 87, or 90 then '1' else '3'.

ref_doc_nbr = For Civil & Military = :obligation.oblg_doc_nbr,
For Revolving = null

trf_date = null

trns_month = null

(c) CMR_4480 table - Only insert if TBO flag = 'Y'.

acpt_date = sysdate

adp_work_code = retrieve adp_wc_d from D_F_FILE where adp_wc_g = :obligation.adp_work_code

amount = :expenditure.amount

batch_nbr = '86'

dist_code = '4'

doc_nbr = generate document number

1st position = K

2nd and 3rd positions = Fiscal month

4th - 9th positions = 022079

object_class_code = :obligation.object_class_code

org_code = '00'

record_code = '02'

ref_doc_nbr = :expenditure.exp_doc_nbr

interface_acpt_flag = 'I'

trns_code = 'PF'

civ_mil_rvolvg_flag = 'M'

trns_type_indic = 'B'

dest = 'A'

other_ref_nbr = Generate reference document number

1st and 2nd positions = :Expenditure.cycle_nbr

3rd - 5th positions = 090

6th - 9th positions = :Expenditure.dssn_nbr

pay_coll_code = '1'

(d) DISBURSEMENT table - Only insert if TBO flag = 'Y'

amount = :expenditure.amount

disb_date = sysdate

disb_doc_nbr = generate Disbursement Number

1st position = K

2nd and 3rd positions = Fiscal month

4th - 9th positions = 022070

exp_doc_nbr = :expenditure.exp_doc_nbr

(4) Updates:

(a) OBLIGATION table

net_unliq_bal = net_unliq_bal - :expenditure.amount

(b) CONTRACT table

ret_perc_doc_nbr = :expenditure.exp_doc_nbr

3.31 V48_INP

A. **Program Name**: V48_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Adjust expenditure amount for a specific retained percent document.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter an expenditure document number to be adjusted. Select and display the description, expenditure amount, and vendor code from the EXPENDITURE table based on the expenditure document number entered and where the status is not equal to 'X' (Canceled). Select and display the obligation document number, net unliquidated balance, adp work code, and vendor code from the OBLIGATION table based on the obligation document number retrieved from the EXPENDITURE table. Select and display the vendor name from the VENDOR table based on the vendor code retrieved from the OBLIGATION table. Select and display the disbursement amount from the DISBURSEMENT table based on the expenditure document number entered.

Prompt user to enter the adjustment amount. Check to make sure the amount of the adjustment + the expenditure amount is greater than or equal to 0. Also check to make sure the adjustment amount has not exceeded the net unliquidated balance for the obligation. If adjustment amount meets the above criteria, update the net unliquidated balance in the OBLIGATION table by the amount of the adjustment.

If the interface acceptance flag = 'I' (in progress) in the CMR_4480 table for the expenditure document number being adjusted, update CMR_4480 and set the amount = amount + adjustment amount.

If the interface acceptance flag <> 'I' then insert a record into the CMR_4480 table for the expenditure document number only if Civil or Military. Insert into CMR_4480 the following:

1. acceptance date = sysdate
2. adp work code = adp work code from EXPENDITURE table

3. amount = adjustment amount
4. batch number =

   Civil & Positive = 55

   Civil & Negative = 52

   Military & Positive = 60

   Military & Negative = 57
5. district code = '4'
6. document number = expenditure document number
7. item code = item code from OBLIGATION table
8. object class code = object class code from OBLIGATION table
9. organization code = organization code from OBLIGATION table
10. pay collect code =

    Civil = null

    Military or Revolving Fund =

        If 2nd and 3rd positions of the obligation document number = '47','52','87','90','92',

        the pay collect code = '3' else the pay collect code = '1'
11. record code = '02'
12. reference document number = obligation document number
13. interface acceptance flag = 'I' (in progress)
14. transaction code =

    Civil = 'MA', Military = 'ME'
15. civil, military, revolving flag =

    Civil = 'C', Military = 'M', Revolving Fund = 'R'
16. transaction type indicator = 'E' (Expenditure)
17. destination = 'A' (Accounting system)

3.32 MAET_INP

A. **Program Name**: MAET_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Maintain the accounting element type table.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter all or a portion of the item code to be entered or edited. Check to see if the item code already exists in the ACCT_ELM_TYPE table.

If already exists, allow user to modify the item code description, civil flag, military flag, and revolving flag. Update the ACCT_ELM_TYPE table with the new changes.

If item code does not exist, the new item code must be in the range between '001' and '499'. Provide user a create record key to add a new item code to the ACCT_ELM_TYPE table. Allow user to enter the description and the appropriate civil flag, military flag, and revolving flag. The only valid entries for the flags are 'Y' (Yes) or null. Insert the record into the ACCT_ELM_TYPE table.

3.33 MAPP_INP

A. **Program Name**: MAPP_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Maintain the appropriation table.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Query and display the civil, military, revolving flag, local appropriation number, and description for all records in the APPROPRIATION table. If editing, allow user to scroll up or down to the appropriate record and edit the appropriation title. Update the changes to the APPROPRIATION table. Provide user a create record key to add a new appropriation code to the APPROPRIATION table.

Revolving Fund appropriation codes cannot be added. Insert the civil, military, revolving flag (Civil = 'C', Military = 'M'), local appropriation number, and description into the APPROPRIATION table.

3.34 MCOO_INP

A. **Program Name**: MCOO_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Maintain obligation classes.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Query and display the description, civil class, military class, and revolving class for all records in the CLASS_OF_OBLIGATION table. If editing, allow user to scroll up or down to the appropriate record and edit the description, civil class, military class, and revolving class. When editing a class, check to make sure the updated class is numeric, and that it does not already exist in another class. Update the changes to the CLASS_OF_OBLIGATION table. Provide user a create record key to add a new obligation class to the CLASS_OF_OBLIGATION table. Insert the description, civil class, military class, and revolving class into the CLASS_OF_OBLIGATION table.

3.35 MEMP_INP

A. **Program Name**: MEMP_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Maintain employee information.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter all or a portion of the employee userid to be entered or edited. Select and display records containing the employee userid, employee name, social security number, and organization code from the EMPLOYEE table based on the employee userid entered.

If editing, allow user to modify the employee name and organization code only. If modifying the organization code, validate against the ORGANIZATION table. Update the EMPLOYEE table with the new changes.

If adding a new employee, provide user a create record key to insert the employee userid, employee name, social security number, and organization code into the EMPLOYEE table. Check to make sure the employee userid and social security number do not already exist in the EMPLOYEE table. Also validate the organization code against the ORGANIZATION table.

3.36 MOCC_INP

A. **Program Name**: MOCC_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Maintain object class code information.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter all or a portion of the object class code to be entered or edited. Check to see if the object class code already exists in the OBJECT_CLASS_CODES table. If already exists, allow user to modify the description, civil flag, military flag, and revolving flag. Civil and Revolving Fund have the same object class codes, but Military does not. Update the OBJECT_CLASS_CODES table with the new changes.

If object class code does not exist, provide user a create record key to add a new object class code to the OBJECT_CLASS_CODES table. Allow user to enter the description and the appropriate civil flag, military flag, and revolving flag. The only valid entries for the flags are 'Y' (Yes) or null. Check to make sure that Military does not mix object class codes with Civil and Revolving Fund. Insert the record into the OBJECT_CLASS_CODES table.

3.37 MORG_INP

A. **Program Name**: MORG_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Maintain organization codes.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter all or a portion of the organization code to be entered or edited. Check to see if the organization code already exists in the ORGANIZATION table. If already exists, allow user to modify the organization name and parent code. The parent code cannot be the same as the organization code. Update the ORGANIZATION table with the new changes.

If organization code does not exist, provide user a create record key to add a new organization code to the ORGANIZATION table. Allow user to enter the new organization code, organization name, and parent code. The parent code cannot be the same as the organization code. Insert the record into the ORGANIZATION table.

3.38 MVEND_INP

A. **Program Name**: MVEND_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Maintain vendor and vendor address information.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt user to enter a vendor code or select a vendor code from an available list. Check to see of the vendor code exists in the VENDOR table.

If vendor already exists, display the vendor name, vendor type, and vendor tax id. Allow user to update any one of these fields. Update the new changes to the VENDOR table. Also query vendor address information for the vendor code entered. Display the vendor address code, vendor address1, vendor address2, vendor city, vendor state, vendor zipcode, vendor country, and vendor phone from the VENDOR_ADDR table based on the vendor code entered. Allow the user to update all fields but the vendor address code, and also provide a create record key to add a new address for the same vendor if necessary. The system generates the next vendor address code by adding 1 to the maximum vendor address code. Update or insert the appropriate vendor address information.

If a vendor code does not exist, provide the user a create record key to add a new vendor. Allow user to enter the vendor name, vendor type, and vendor tax id. Insert the new record into the VENDOR table. The user must also insert vendor address information for the new vendor. The system will automatically assign an '01' for the first vendor address code. Insert the vendor address information into the VENDOR_ADDR table.

3.39 MVIS_INP

A. **Program Name**: MVIS_INP

B. **Program Language**: SQL*Forms

C. **Purpose**: Used mostly for conversion purposes. Allows user to fix certain fields that may not be correct.

D. **Entry Point**: Executed through VIS menu.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Prompt the user to enter the obligation document number. The obligation document number must exist in the OBLIGATION table and the status = 'C'. The system retrieves and displays the contract number, contract modification number, contract amount, vendor code, commitment document number, adp work code, local appropriation number, object class code, item code, and organization code from the OBLIGATION table based on the obligation document number entered. The vendor name is retrieved from the VENDOR table based on the vendor code retrieved from the OBLIGATION table.

The contract modification number can be updated if = -1. After a new contract modification number is entered, contract details must be entered. Allow the user to enter the contract reference modification number, amount, date of award, ending date, FOB date, tax ID, and contract officer representative. Check to make sure the date of award <= ending date. Default the ending date into the FOB date.

If the user changes the vendor code, check to make sure the vendor code exists in the vendor table. If the adp work code is changed, the 1st digit should be between A-I for Civil, J-T for Military, and V for Revolving Fund. If the adp work code is a Military work code, validate against the D_F_FILE table. If the local appropriation number is changed, validate against the APPROPRIATION table based on the civil, military, revolving flag.

If the object class code is changed, validate against the OBJECT_CLASS_CODES table that the code is valid based on the civil, military, revolving flag. If the item code is changed, validate against the ACCT_ELM_TYPE table. If civil or revolving fund, code must be between 202 and

399. If military, code must be between 202 and 399, 500. If revolving fund prip, code must be 474, 477, 479, or 498. If the organization code is modified, validate against the ORGANIZATION table.

Once all changes have been made, update the OBLIGATION, CONTRACT, EXPENDITURE, SCHEDULED_PAYMNT, and INVOICE tables with the appropriate changes.

4. SQL*Plus

4.1 CHTRV_SQL

A. **Program Name**: CHTRV_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Script is used when processing IATS travel cost information. It displays all records ready to be processed that match to an obligation. This file should be scanned to find any multiple obligations for one travel order number.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: N/A

F. **Output**: Script file displays the travel order number, vendor code, commitment document number, obligation document number and amount from the TRAVEL_EXP and OBLIGATION tables for the records ready to be processed that match to an obligation.

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select the travel order number, vendor code, commitment document number, obligation document number and amount from the TRAVEL_EXP and OBLIGATION tables where the travel order number in the TRAVEL_EXP table matches the commitment document number in the OBLIGATION table, the 2nd and 3rd positions of the obligation document number are '80', '81', or '82' which identifies the records as travel, the acceptance flag in the TRAVEL_EXP table is 'I' for in progress, and the length of the travel order number in the TRAVEL_EXP table is 6 positions. The records are ordered by the travel order number.

4.2 COEMIS_SQL

A. **Program Name**: COEMIS_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Batch transactions to be updated into COEMIS.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: Enter the transaction month (i.e. 01 for January) and the transfer date (i.e. 10-JAN-92) when prompted.

F. **Output**: Create output file CMR4480_LST containing the selected transactions in COEMIS format.

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select records containing the district code, organization code, transaction date, batch number, record code, pay collect code, adp work code, object class code, amount, document number, reference document number, and other reference number from the CMR_4480 table where the interface acceptance flag = 'I' (in progress) and spool them into a file called CMR4480_LST. The CMR4480 table is then updated setting the transaction month to the month that was entered, the transfer date to the date that was entered, and the interface acceptance flag to 'A' for all records that met the selection criteria.

4.3 DEOBLG_SQL

A. **Program Name**: DEOBLG_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Generate a listing of possible obligations to deobligate.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: N/A

F. **Output**: File (DEOBLG_LST) containing obligation document numbers.

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select obligation document numbers from the OBLIGATION table where the obligation document number in the OBLIGATION table matches the obligation document number in the EXPENDITURE table, and the partial final flag = 'F', and the net unliquidated balance <> 0, and the scheduled payment date for the expenditure is not null.

I. **Other Program Notes**: The records are ordered by vendor name, contract number, and contract modification number.

4.4 DEOBLIGATE_SQL

A. **Program Name**: DEOBLIGATE_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Generate a report of possible obligations to deobligate.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: N/A

F. **Output**: File (DEOBLIGATE_RPT) containing obligation information.

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select the contract number, contract modification number, obligation document number, commitment document number, vendor name, net unliquidated balance, scheduled payment date, and expenditure document number from the OBLIGATION, EXPENDITURE, and VENDOR tables where the obligation document number in the OBLIGATION table matches the obligation document number in the EXPENDITURE table, and the partial final flag = 'F', and the net unliquidated balance <> 0 and the scheduled payment date for the expenditure is not null.

I. **Other Program Notes**: The records are ordered by vendor name, contract number and contract modification number.

4.5 DISBURSEMENT_SQL

A. **Program Name**: DISBURSEMENT_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Generate a summary disbursement report by vendor.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: N/A

F. **Output**: File (DISBURSEMENT_RPT) containing summary disbursement information.

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select records containing the vendor name, expenditure document number, partial pay number, amount, contract number, invoice number, scheduled payment date, disbursement date, DOV number, and check number from the DISBURSEMENT, EXPENDITURE, VENDOR and SCHEDULED_PAYMNT tables where the expenditure document number in the EXPENDITURE table matches the expenditure document number in the DISBURSEMENT table, the vendor code in the SCHEDULED_PAYMNT table matches the vendor code in the VENDOR table, the expenditure document number in the SCHEDULED_PAYMNT table matches the expenditure document number in the DISBURSEMENT table, the amount in the SCHEDULED_PAYMNT table matches the amount in the in the DISBURSEMENT table, and the disbursement date is greater than '30-SEP-91'.

I. **Other Program Notes**: The records are ordered by the vendor name and the disbursement date.

4.6 EXPFILE

A. **Program Name**: EXPFILE

B. **Program Language**: SQL*Plus

C. **Purpose**: Export the VIS database.

D. **Entry Point**: Executed from the batch procedure EXP_BAT.

E. **Input Data Requirements**: N/A

F. **Output**: Export file (i.e. VIS0131_DMP).

G. **Abnormal Termination**: N/A

H. **Program Execution**: Export the VIS database with the following options:

FULL=N

FILE=VIS0131_DMP (this filename should be changed to the appropriate date)

GRANTS=Y

INDEXES=N

OWNER=VIS

COMPRESS=N

4.7 FIXPAYCOLL_SQL

A. **Program Name**: FIXPAYCOLL_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Change pay collect code for disbursement transactions to the same pay collect code as the matching expenditure transactions after uploading the disbursement transactions from the check writing routine to the VIS data base.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Update the pay collect code of the disbursement transactions in the CMR_4480 table to the same pay collect code of the matching expenditure transactions where the Civil, Military, Revolving flag = 'M', the transaction code = 'PA', and the transaction date = the current date.

4.8 PRE1166_SQL

A. **Program Name**: PRE1166_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Generate a preliminary 1166 for the selected date for commercial vendors.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: Enter the scheduled payment date (i.e. 01-FEB-92) when prompted.

F. **Output**: Create output file PRE1166_LST.

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select records containing the vendor code, vendor name, expenditure document number, local appropriation number, object class code, adp work code, employee name, scheduled payment date, and amount from the SCHEDULED_PAYMNT, VENDOR, and EMPLOYEE tables where the scheduled payment date is less than or equal to the scheduled payment date entered, the vendor code in the VENDOR_CODE table matches the vendor code in the SCHEDULED_PAYMNT table, the employee userid in the EMPLOYEE table matches the initial entry id in the SCHEDULED_PAYMNT table, the transfer date is null, and the initial entry id is not like 'ADV%'.

I. **Other Program Notes**: The records are ordered by the vendor name and the adp work code. Subtotals are computed by vendor as well as a grand total.

4.9 PRE1166TRV_SQL

A. **Program Name**: PRE1166TRV_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Generate a preliminary 1166 for the selected date for travelers.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: Enter the scheduled payment date (i.e. 01-FEB-92) when prompted.

F. **Output**: Create output file PRE1166TRV_LST.

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select records containing the vendor name, vendor code, expenditure document number, local appropriation number, object class code, adp work code, amount, initial entry id, and scheduled payment date from the SCHEDULED_PAYMNT and VENDOR tables where the scheduled payment date is less than or equal to the scheduled payment date entered, the vendor code in the VENDOR_CODE table matches the vendor code in the SCHEDULED_PAYMNT table, the transfer date is null, and the initial entry id is like 'ADV%'.

I. **Other Program Notes**: The records are ordered by the vendor name and the adp work code.

4.10 PREV23RPT_SQL

A. **Program Name**: PREV23RPT_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Generate a report of those travel order numbers that do not match to an obligation document number. It also generates a list of those travel order numbers where the vendor does not exist.

D. **Entry Point**: Executed from the batch procedure TRAVEL_BAT.

E. **Input Data Requirements**: N/A

F. **Output**: Create output file TRAVEL_REJ.

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select records containing travel order number, vendor code, and vendor name from the TRAVEL_EXP and VENDOR tables where the vendor code in the VENDOR table matches the vendor code in the TRAVEL_EXP table, the travel order number is not like 'E%' and the acceptance flag = 'I'. This script also selects records containing the travel order number and vendor code from the TRAVEL_EXP table where the vendor code does not exist in the VENDOR table and the acceptance flag = 'I'.

The TRAVEL_EXP table is then updated setting the acceptance flag = 'R' for those records that met the select criteria.

4.11 TBO_SQL

A. **Program Name**: TBO_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Generate a TBO report for a specified date range.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: Enter the start date and the end date when prompted.

F. **Output**: Create output file TBO_RPT.

G. **Abnormal Termination**: N/A

H. **Program Execution**: Select records containing the dssn number, cycle number, adp work code, obligation document number, expenditure document number, disbursement document number, amount, and local appropriation number from the DISBURSEMENT, EXPENDITURE, and OBLIGATION tables where the entry date in the EXPENDITURE table is between the start date entered and the end date entered, the expenditure document number in the EXPENDITURE table matches the expenditure document number in the DISBURSEMENT table, the obligation document number in the OBLIGATION table matches the obligation document number in the EXPENDITURE table, and the dssn number is not null.

I. **Other Program Notes**: The records are ordered by dssn number, cycle number, and local appropriation number.

4.12 UPOBLG_SQL

A. **Program Name**: UPOBLG_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Convert the IATS travel order number to an obligation document number for use by V23 (Process IATS Travel Cost Information).

D. **Entry Point**: Executed from the batch procedure TRAVEL_BAT.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Update the TRAVEL_EXP table setting the travel order number in the TRAVEL_EXP table to the obligation document number in the OBLIGATION table where the travel order number in the TRAVEL_EXP table matches the commitment document number in the OBLIGATION table, the 2nd and 3rd positions of the obligation document number are '80', '81', or '82' which identifies the records as travel, the vendor code in the OBLIGATION table matches the vendor code in the TRAVEL_EXP table, the acceptance flag in the TRAVEL_EXP table is 'I' for in progress, the travel order number in the TRAVEL_EXP table is not like 'E%', and the length of the travel order number in the TRAVEL_EXP table is 6 positions.

4.13 UPVEND_SQL

A. **Program Name**: UPVEND_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Add a 0 to the beginning of the vendor code in the TRAVEL_EXP table.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**: Update the TRAVEL_EXP table adding a 0 to the beginning of the vendor code where the acceptance flag = 'I' and the length of the vendor code is 9 positions.

4.14 V27_SQL

A. **Program Name**: V27_SQL

B. **Program Language**: SQL*Plus

C. **Purpose**: Create a file of scheduled payments for commercial vendors and travelers for the selected date.

D. **Entry Point**: Login to SQL*Plus with the appropriate VIS userid/password.

E. **Input Data Requirements**: Enter the scheduled payment date (i.e. 01-FEB-92) when prompted.

F. **Abnormal Termination**: N/A

G. **Program Execution**:

For commercial vendors: select records containing the district code, expenditure document number, local appropriation number, ams code for Military, vendor code, vendor name, vendor address 1 and 2, vendor city, vendor state, vendor zipcode, scheduled payment date, amount, object class code, adp work code, employee name, and allotment for Military from the SCHEDULED_PAYMNT, VENDOR, VENDOR_ADDR, EMPLOYEE, and D_F_FILE where the scheduled payment date in the SCHEDULED_PAYMNT table is less than or equal to the scheduled payment date entered, the transfer date in the SCHEDULED_PAYMNT table is null, the vendor code in the SCHEDULED_PAYMNT table matches the vendor code in the VENDOR table, the vendor code in the VENDOR_ADDR table matches the vendor code in the SCHEDULED_PAYMNT table, the vendor address code in the VENDOR_ADDR table matches the vendor address code in the VENDOR_ADDR table, the employee userid in the EMPLOYEE table matches the initial entry id in the SCHEDULED_PAYMNT table, and if Military the D level adp work code in the D_F_FILE matches the adp work code in the SCHEDULED_PAYMNT table. The records are ordered by local appropriation number and vendor code.

For travelers: select records containing the district code, expenditure document number, local appropriation number, ams code for Military, vendor code, vendor name, vendor address 1 and 2, vendor city, vendor state, vendor zipcode, scheduled payment date, amount, object class code, adp work code, initial entry id, and allotment for Military from the SCHEDULED_PAYMNT,

VENDOR, VENDOR_ADDR, and D_F_FILE where the scheduled payment date in the SCHEDULED_PAYMNT table is less than or equal to the scheduled payment date entered, the transfer date in the SCHEDULED_PAYMNT table is null, the vendor code in the SCHEDULED_PAYMNT table matches the vendor code in the VENDOR table, the vendor code in the VENDOR_ADDR table matches the vendor code in the SCHEDULED_PAYMNT table, the vendor address code in the VENDOR_ADDR table matches the vendor address code in the VENDOR_ADDR table, the initial entry id in the SCHEDULED_PAYMNT table is like 'ADV%', and if Military the D level adp work code in the D_F_FILE matches the adp work code in the SCHEDULED_PAYMNT table. The records are ordered by local appropriation number and vendor code.

Update the SCHEDULED_PAYMNT table setting the transfer date = sysdate for those records that met the selection criteria.

5. SQL*Loader

5.1 V22_CTL

A. **Program Name**: V22_CTL

B. **Program Language**: SQL*Loader

C. **Purpose**: Load IATS travel cost information into the TRAVEL_EXP table.

D. **Entry Point**: Login to SQL*Loader with the appropriate VIS userid/password.

E. **Input Data Requirements**: Input file IATS_DAT.

F. **Output**: N/A

G. **Abnormal Termination**: Capture errors in logfile V22_LOG.

H. **Program Execution**: Load vendor code, obligation document number, expenditure amount, advance amount, acceptance flag, and transfer date into the TRAVEL_EXP table.

5.2 V28A_CTL

A. **Program Name**: V28A_CTL

B. **Program Language**: SQL*Loader

C. **Purpose**: Load disbursement information into the DISBURSEMENT table.

D. **Entry Point**: Login to SQL*Loader with the appropriate VIS userid/password.

E. **Input Data Requirements**: Input file V28A_DAT.

F. **Output**: N/A

G. **Abnormal Termination**: Capture errors in logfile V28A_LOG.

H. **Program Execution**: Load disbursement document number, amount, check number, DOV number, disbursement date, and expenditure document number into the DISBURSEMENT table.

5.3 V28B_CTL

A. **Program Name**: V28B_CTL

B. **Program Language**: SQL*Loader

C. **Purpose**: Load PA transaction information into the CMR_4480 table.

D. **Entry Point**: Login to SQL*Loader with the appropriate VIS userid/password.

E. **Input Data Requirements**: Input file V28B_DAT.

F. **Output**: N/A

G. **Abnormal Termination**: Capture errors in logfile V28B_LOG.

H. **Program Execution**: Load district code, organization code, acceptance date, batch number, record code, transaction code, pay collect code, adp work code, object class code, amount, document number, reference document number, civil, military and revolving flag, destination, interface acceptance flag and transaction type indicator into the CMR_4480 table.

5.4 V44_CTL

A. **Program Name**: V44_CTL

B. **Program Language**: SQL*Loader

C. **Purpose**: Load Citicorp information into the AUTO_SCHED table.

D. **Entry Point**: Login to SQL*Loader with the appropriate VIS userid/password.

E. **Input Data Requirements**: Input file AUTO_DAT.

F. **Output**: N/A

G. **Abnormal Termination**: Capture errors in logfile V44_LOG.

H. **Program Execution**: Load obligation document number, amount, vendor code, invoice number, scheduled payment date, transfer date, description, and acceptance flag into the AUTO_SCHED table.

5.5 VISCIV_CTL

A. **Program Name**: VISCIV_CTL

B. **Program Language**: SQL*Loader

C. **Purpose**: Load Civil and Revolving Fund adp work codes and appropriation numbers into the CIVRF table.

D. **Entry Point**: Login to SQL*Loader with the appropriate VIS userid/password.

E. **Input Data Requirements**: Input file CIVRF_DAT.

F. **Output**: N/A

G. **Abnormal Termination**: Capture errors in logfile VISCIV_LOG.

H. **Program Execution**: Load adp work code and local appropriation number into the CIVRF table.

5.6 VISDF_CTL

A. **Program Name**: VISDF_CTL

B. **Program Language**: SQL*Loader

C. **Purpose**: Load D, F, and G level adp work codes, ams codes, local appropriation numbers, and allotments for Military into the D_F_FILE table.

D. **Entry Point**: Login to SQL*Loader with the appropriate VIS userid/password.

E. **Input Data Requirements**: Input file D_T_F_DAT.

F. **Output**: N/A

G. **Abnormal Termination**: Capture errors in logfile VISDF_LOG.

H. **Program Execution**: Load G level adp work code, F level adp work code, D level adp work code, ams code, allotment, and local appropriation number into the D_F_FILE table.

6. SCL PROCEDURES

6.1 BATCH_TRV

A. **Program Name**: BATCH_TRV

B. **Program Language**: SCL

C. **Purpose**: Run the PRO*C programs V23 (Process IATS Travel Cost Information) and V31 (Process Recurring Expenditures).

D. **Entry Point**: Executed from the batch procedure TRAVEL_BAT.

E. **Input Data Requirements**: N/A

F. **Output**: Create output files V23_RPT and V31_RPT.

G. **Abnormal Termination**: N/A

H. **Program Execution**:

1. Create file connection to $output for V31_RPT
2. Run V31
3. Delete file connection to $output for V31_RPT
4. Create file connection to $output for V23_RPT
5. Run V23
6. Delete file connection to $output for V23_RPT

6.2 CC_BAT

A. **Program Name**: CC_BAT

B. **Program Language**: SCL

C. **Purpose**: Run the PRO*C program V42 (Process CITICORP Information).

D. **Entry Point**: Submit job from the CDC prompt.

E. **Input Data Requirements**: N/A

F. **Output**: Create output file V42_RPT.

G. **Abnormal Termination**: N/A

H. **Program Execution**:

1. Login U4RMOFA, identifying adp work code and job type
2. Set ORACLE environment to sid s
3. Change to VSDBA subcatalog
4. Run V42
5. Print V42_RPT
6. Logout

6.3 DEOBLG_BAT

A. **Program Name**: DEOBLG_BAT

B. **Program Language**: SCL

C. **Purpose**: Run the PRO*C program V45 (Automatic Obligation Liquidation).

D. **Entry Point**: Submit job from CDC prompt.

E. **Input Data Requirements**: N/A

F. **Output**: Create output file V45_RPT.

G. **Abnormal Termination**: N/A

H. **Program Execution**:

1. Login U4RMOFA, identifying adp work code and job type
2. Set ORACLE environment to sid s
3. Change to VSDBA subcatalog
4. Create file connection to $output for V45_RPT
5. Run V45
6. Delete file connection to $output for V45_RPT
7. Logout

6.4 EXP_BAT

A. **Program Name**: EXP_BAT

B. **Program Language**: SCL

C. **Purpose**: Run the batch file EXPFILE.

D. **Entry Point**: Submit job from the CDC prompt.

E. **Input Data Requirements**: N/A

F. **Output**: N/A

G. **Abnormal Termination**: N/A

H. **Program Execution**:

1. Login U4RMOFA, identifying adp work code and job type
2. Set ORACLE environment to sid s
3. Change to VSDBA subcatalog
4. Run export (exp parfile=EXPFILE)
5. Logout

6.5 PAID_MASTER_BAT

A. **Program Name**: PAID_MASTER_BAT

B. **Program Language**: SCL

C. **Purpose**: Run report DISBURSEMENT_SQL.

D. **Entry Point**: Submit job from CDC prompt.

E. **Input Data Requirements**: N/A

F. **Output**: Create output file DISBURSEMENT_RPT.

G. **Abnormal Termination**: N/A

H. **Program Execution**:

1. Login U4RMOFA, identifying adp work code and job type
2. Set ORACLE environment to sid s
3. Change to VSDBA subcatalog
4. Run DISBURSEMENT_SQL
5. Print DISBURSEMENT_RPT
6. Logout

6.6 PARTPAY_BAT

A. **Program Name**: PARTPAY_BAT

B. **Program Language**: SCL

C. **Purpose**: Run report PARTPAY.

D. **Entry Point**: Submit job from CDC prompt.

E. **Input Data Requirements**: N/A

F. **Output**: Create output file PARTPAY_LIS.

G. **Abnormal Termination**: N/A

H. **Program Execution**:

1. Login U4RMOFA, identifying adp work code and job type
2. Set ORACLE environment to sid s
3. Run report PARTPAY
4. Print PARTPAY_LIS
5. Logout

6.7 TRAVEL_BAT

A. **Program Name**: TRAVEL_BAT

B. **Program Language**: SCL

C. **Purpose**: Run UPOBLG_SQL, PREV23RPT_SQL and the batch procedure BATCH_TRV.

D. **Entry Point**: Submit job from the CDC prompt.

E. **Input Data Requirements**: N/A

F. **Output**: Create output files TRAVEL_REJ, V23_RPT and V31_RPT.

G. **Abnormal Termination**: N/A

H. **Program Execution**:

1. Login U4RMOFA, identifying adp work code and job type
2. Set ORACLE environment to sid s
3. Change to VSDBA subcatalog
4. Run UPOBLG_SQL
5. Run PREV23RPT_SQL
6. Execute BATCH_TRV
7. Print TRAVEL_REJ, V23_RPT, and V31_RPT
8. Logout

7. SQL*ReportWriter

7.1 PARTPAY_REP

A. **Program Name**: PARTPAY_REP

B. **Program Language**: SQL*ReportWriter

C. **Purpose**: Generate partial pay sheet.

D. **Entry Point**: Executed as a batch program.

E. **Input Data Requirements**: N/A

F. **Output**: Generate report PARTPAY_LIS.

G. **Abnormal Termination**: N/A

H. **Program Execution**:

1. Select and display the adp work code, obligation document number, obligation amount, net unliquidated balance, obligation entry date, object class code, commitment document number, vendor name, local appropriation number, contract number, contract modification number, contract reference modification number, contract amount, date of award, contract ending date, discount days, discount percent, retained percent, retained percent document number, organization name, organization code from the VENDOR, DISC_TERMS, CONTRACT, OBLIGATION, ORGANIZATION, and EXPENDITURE tables where the obligation status = 'C', the vendor code from the OBLIGATION table matches the vendor code in the VENDOR table, the contract number and contract modification number in the OBLIGATION table matches the contract number and contract modification number in the CONTRACT table, the organization code in the OBLIGATION table matches the organization code in the ORGANIZATION table, the discount terms code in the CONTRACT table matches the discount terms code in the DISC_TERMS table, the net unliquidated balance of the obligation > 0 or the net unliquidated balance = 0 and the scheduled payment date is null.

2. Select and display each expenditure document number, invoice number, expenditure entry date, partial pay number, expenditure amount, and scheduled payment date from the EXPENDITURE table based on the obligation document number displayed in Step 1.

3. Select and display the disbursement document number, DOV number, check number, and amount from the DISBURSEMENT table based on the expenditure document number displayed in Step 2.

8. PRO*C

8.1 V23_PC

A. **Program Name**: V23

B. **Program Language**: PRO*C

C. **Purpose**: The purpose of this program is to query the data base for travel expenditures that should be processed. V23 generates the expenditure transaction, required interface transaction for COEMIS, schedules the expenditure for payment, performs recoupment of advance, if appropriate, and updates the TRAVEL_EXP and OBLIGATION tables consistent with the transactions processed.

D. **Entry Point**: Executed as a batch program. Command line must contain the userid and password for the data base being accessed as well as a maximum allowable error count for abnormal transactions.

E. **Input Data Requirements**: N/A

F. **Output**: The program prints a standard report with header and trailer information containing:

(1) The expenditure and obligation document numbers for each transaction.

(2) The total number of transactions accepted and rejected.

G. **Abnormal Termination**: The program will abnormally terminate upon encountering an ORACLE error. Both a descriptive and the ORACLE error messages are printed. In VIS23 a logical unit of work is defined as 100 transactions. Individual transactions may be rejected within this logical unit of work. If an ORACLE error is encountered in processing a transaction set, i.e. logical unit of work, the entire set of transactions will be rolled back and the program will continue with the next set.

H. **Program Execution**: The program processes transactions based on the TRAVEL_EXP table where the acpt_flag = 'I'. V23 uses a cursor to fetch transactions from the TRAVEL_EXP table according to the criterion stated above. A commitment is made for every logical unit of work (see G above) or at successful termination of the program.

(1) Fetch the following information from TRAVEL_EXP(t) where acpt_flag = 'I':

(a) adv_amt: advance amount, if any, made to the vendor.

(b) exp_amt: amount of the travel expenditure. (NOTE: This amount is calculated as output from the IATS Standard Travel System.)

(c) oblg_doc_nbr: the obligation document number for the travel expenditures.

(d) vendor_code: the vendor code of the traveler.

(2) Retrieve the following information from the OBLIGATION (o) table based on the oblg_doc_nbr in H(1)(c).

(a) item_code: The item code for this obligation.

(b) object_class_code: The object class code for this obligation.

(c) adp_work_code: The COEMIS ADP work code for this obligation.

(d) org_code: The responsible organization code for this obligation.

(e) civ_mil_rvolvg_flag: An indicator of the category of funds for this obligation; civil, military, or revolving fund.

(f) loc_appn_nbr: The appropriation number locally assigned to this obligation.

(g) vendor_code: The vendor code assigned to this obligation.

(h) net_unliq_bal: The net unliquidated balance for this obligation. If an obligation document is not found in the OBLIGATION table, increment the error count and go to step H(1).

(3) Check the obligation document against the following criteria:

(a) Is o.vendor_code = t.vendor_code?

(b) Is o.net_unliq_bal >= t.exp_amt?

If either test fails, update TRAVEL_EXP set t.acpt_flag = 'R' for the current transaction and skip to step H(1).

(4) Create the expenditure document for this transaction by forming a concatenated string consisting of:

(a) char 1 - 'F'.

(b) char 2-3 - Fiscal month of the system date.

(c) char 4-5 - Fiscal year of the system date.

(d) char 6-9 - Sequence number for expenditure documents (from RDBMS)

(5) Insert the following record into the EXPENDITURE table:

amount = t.exp_amt

entry_date = sysdate

exp_doc_nbr = document number generated in step 4

init_entry_id = 'Travel Exp'

oblg_doc_nbr = t.oblg_doc_nbr

partial_final_flag = 'p'

vendor_addr_code = '01'

vendor_code = t.vendor_code

adp_work_code = t.adp_work_code

status = 'C'

cycle_nbr = null

acpt_date = sysdate

del_date = sysdate

descr1 = 'VIS generated travel expenditure'

descr2 = null

disc_terms_code = null

dssn_nbr = null

inv_nbr = null

pp_nbr = 0

scheduled_payment_date = next business day based on system time when V23 is executed. If this date is a Friday or Saturday, schedule expenditure for Monday.

tbo_flag = null

(6) Create the interface transaction by inserting the following record into the CMR_4480 table:

acpt_date = sysdate

adp_work_code = o.adp_work_code

amount = t.exp_amt

batch_nbr = '52' for civil, '57' for military, and '47' for revolving fund

civ_mil_rvolvg_flag = o.civ_mil_rvolvg_flag

dest = 'A'

dist_code = '4' (Note: This is code for WES.)

doc_nbr = expenditure document number generated in step 4

interface_acpt_flag = 'I'

record_code = '02'

trns_code = 'MA' for civil or military and 'MD' for revolving fund

trns_type_indic = 'E'

item_code = o.item_code

object_class_code = o.object_class_code

org_code = o.org_code

other_ref_nbr = null

pay_coll_code = null for civil and '3' for military or revolving fund

ref_doc_nbr = a.oblg_doc_nbr for civil or military and null for revolving fund

trf_date = null

trns_month = null

(7) Schedule the expenditure for payment by inserting the following record into the SCHEDULED_PAY table as follows:

adp_work_code = For o.civil_mil_rvolvg_flag = 'C' = first 5 characters of o.adp_work_code concatenated with the string '0000000000'. For o.civil_mil_rvolvg_flag = 'M' = D level adp work code retrieved from the D_F_FILE table. For o.civil_mil_rvolvg_flag = 'R' = 'VW0000000000000'.

amount = t.exp_amt

exp_doc_nbr = exp_doc_nbr generated in step H(4)

init_entry_id = The entry_id is constructed from the base character string of "ADV000000000". IF the t.adv_amt > t.exp_amt use t.adv_amt, else use t.exp_amt. Copy the appropriate amount represented as a character string into the base string, right justified.

loc_appn_nbr = o.loc_appn_nbr

object_class_code = o.object_class_code

scheduled_payment_date = scheduled payment date generated in step H(5)(t)

status = 'C'

vendor_code = t.vendor_code

vendor_addr_code = '01'

contr_mod_nbr = null

contr_nbr = null

trns_date = sysdate

trnsf_date = null

(8) If t.adv_amt > 0 and o.civ_mil_rvolvg_flag = 'M', generate the appropriate transaction to schedule recoupment of the advance amount by inserting a record into the SCHEDULE_PAYMNT table as follows:

adp_work_code = D level adp_work_code as determined in step H(7)(a)(2)

amount = t.adv_amt or t.exp_amt if t.adv_amt > t.exp_amt

exp_doc_nbr = document number as generated in step H(4) for matching transaction

init_entry_id = "ADV000000000"

loc_appn_nbr = o.loc_appn_nbr

object_class_code = o.object_class_code

scheduled_payment_date = date as determined in step H(5)

status = 'I'

vendor_code = '0000000000'

vendor_addr_code = '01'

contr_mod_nbr = null

contr_nbr = null

trns_date = sysdate

trnsf_date = null

(9) Set t.acpt_flag = 'A' for current entry being processed in TRAVEL_EXP.

(10) Update the OBLIGATION table, set net_unliq_amt = net_unliq_bal - t.exp_amt where o.oblg_doc_nbr = current oblg_doc_nbr.

(11) If 100 transactions have been processed, commit work to data base and go to step (1), else go to step (1).

(12) If no records are fetched from cursor, commit outstanding transactions, if any, print summary report, and exit from program.

I. **Other Program Notes**:

(1) District Code used in step H(6) is specific for the installation. The parameter will have to be modified and separate programs maintained if the district code changes.

(2) The expenditure document number generated in step H(4) required the existence of an ORACLE sequence number named EXPSEQ in the data base.

(3) V23 locks the tables EXPENDITURES, SCHEDULED_PAYMENT, OBLIGATION, TRAVEL_EXP, D_F_FILE, and CMR_4480 in exclusive mode. This program should be run alone to avoid a possible deadlock situation.

8.2 V31_PC

A. **Program Name**: V31

B. **Program Language**: PRO*C

C. **Purpose**: This program queries a VIS data base to determine if a recurring expenditure document is due for processing. If so, the program generates the appropriate expenditure transaction for insertion into the VIS data base plus the accompanying transaction for subsequent processing by COEMIS.

D. **Entry Point**: Executed as a batch program. Command line must contain the userid and password for the data base being processed.

E. **Input Data Requirements**: N/A

F. **Output**: Program prints the following report:

(1) Listing of the expenditure document number, amount, and referenced obligation document number for each expenditure transaction generated.

(2) Total number of transactions processed.

G. **Abnormal Termination**: The program will abnormally terminate upon encountering an ORACLE error. Both a descriptive and the ORACLE error messages are printed. The current Transaction is rolled back and the program exited. Important Note: The program performs only one commit, after processing all transactions; consequently, an error in any transaction will roll back all previously completed transactions.

H. **Program Execution**: The program processes transactions based on records in the RECURRING_EXP table.

(1) The records in the RECURRING_EXP table must meet the following criteria:

(a) Status = 'A'

(b) Next expenditure date <= system date

(c) Next expenditure date <= expiration date

(2) Fetch the following information for each recurring expenditure to be processed from the RECURRING_EXP (re) table:

(a) oblg_doc_nbr: The obligation document to be referenced by the expenditure.

(b) init_entry_id: The userid of the person who created this recurring expenditure

record.

(c) amount: The amount of the expenditure transaction.

(d) object_class_code: The object class code for expenditure transaction.

(e) descr1: The first line of the description entered for this expenditure.

(f) descr2: The second line of the description entered for this expenditure.

(g) vendor_code: The vendor code for this expenditure.

(h) item_code: The item code for this expenditure.

(i) vendor_addr_code: The vendor address code for this expenditure.

(j) paymnt_period: The payment period, in months, covered by this expenditure.

(k) adp_work_code: The COEMIS ADP work code for this expenditure.

(2) If no records are found, all transactions have been processed, print final report, commit, and exit program.

(3) Check the information from H(2) against the OBLIGATION table.

(a) Is obligation re.oblg_doc_nbr certified?

(b) Is the net unliquidated balance for re.oblg_doc_nbr >= re.amount? If either of these tests fail, print a descriptive error message, and go to step 1, i.e., skip this transaction. Else retrieve the following information from the OBLIGATION (o) table for re.oblg_doc_nbr.

(c) org_code: The responsible organization code for this obligation.

(d) civ_mil_rvolvg_flag: An indicator of the category of funds for this obligation, i.e., civil, military, or revolving fund.

(4) Create the expenditure document for this transaction by forming a concatenated string consisting of:

(a) char 1 - 'F'

(b) char 2-3 - Fiscal month of the system date

(c) char 4-5 - Fiscal year of the system date

(d) char 6-9 - Sequence number for expenditure documents (from RDBMS)

(5) Insert the following record into the EXPENDITURE table:

amount = re.amount

entry_date = sysdate

exp_doc_nbr = document number generated in step 4

init_entry_id = re.init_entry_id

oblg_doc_nbr = re.oblg_doc_nbr

partial_final_flag = 'P'

vendor_addr_code = re.vendor_addr_code

vendor_code = re.vendor_code

adp_work_code = re.adp_work_code

status = 'C'

acpt_date = sysdate

del_date = sysdate

descr1 = re.descr1

descr2 = re.descr2

dssn_nbr = null

inv_nbr = null

pp_nbr = the next sequential partial pay number for all expenditures in the EXPENDITURE table referencing this re.oblg_doc_nbr

scheduled_paymnt_date = null

tbo_flag = null

(6) Create the interface transaction by inserting the following record into the CMR_4480 table:

acpt_date = sysdate

adp_work_code = re.adp_work_code

amount = re.amount

batch_nbr = '52' for civil, '57' for military, and '47' for revolving fund

civ_mil_rvolvg_flag = o.civ_mil_rvolvg_flag

dest = 'A'

dist_code = '4' (Note: This is code for WES)

doc_nbr = expenditure document number generated in step 4

interface_acpt_flag = 'I'

record_code = '02'

trns_code = 'MA' for civil or military and 'MD' for revolving fund

trns_type_indic = 'E'

item_code = re.item_code

object_class_code = re.object_class_code

org_code = o.org_code

other_ref_nbr = null

pay_coll_code = null for civil and '3' for military or revolving fund

ref_doc_nbr = re.oblg_doc_nbr for civil or military and null for revolving fund

trf_date = null

trns_month = null

(7) Update the RECURRING_EXP table for the record currently being processed and add re.paymnt period to re.next_exp_date.

(8) Update the OBLIGATION table for document number re.oblg_doc_nbr decreasing o.net_unliq_bal by re.amount.

(9) Go to Step 1 and repeat.

I. **Other Program Notes**:

(1) District code used in step H(6) is specific for the installation. This attribute will have to be modified if the district code changes.

(2) The expenditure document number generated in step H(4) requires the existence of an ORACLE sequence number names EXPSEQ in the database.

(3) This program locks the following tables in exclusive mode during executions: EXPENDITURE, RECURRING_EXP, CMR_4480, and OBLIGATION. Care should be taken that VIS31 is run alone to avoid a possible deadlock situation.

8.3 V42_PC

A. **Program Name**: V42

B. **Program Language**: PRO*C

C. **Purpose**: The purpose of this program is to query the data base for a set of expenditures to be scheduled for payment. V42 generates the required expenditure document and interface transaction for COEMIS, schedules the expenditures for payment, and updates the AUTO_SCHED, OBLIGATION, and INVOICE tables consistent with the transactions processed. Expenditures to be scheduled are flagged in the AUTO_SCHED table.

D. **Entry Point**: Executed as a batch program. Command line must contain the userid and password for the data base being processed as well as the file name (complete path) for the output report.

E. **Input Data Requirements**: N/A

F. **Output**: The program prints (to the output file specified on the command line) a report with header and trailer information containing:

(1) The expenditure and obligation document numbers for each transaction processed.

(2) The total number of transactions accepted and rejected.

G. **Abnormal Termination**: The program will abnormally terminate upon encountering an ORACLE error. Both a descriptive and the ORACLE error messages are printed. The current transaction is rolled back and the program exited. Important Note: See Section 4 for description of commit interval used by VIS42. An error will affect all transactions for a logical unit of work as defined for VIS42.

H. **Program Execution**: The program processes transactions based on the AUTO_SCHED table where the acpt_flag = 'I'. V42 uses nested cursors to group expenditures by vendor code and invoice number and these two variables drive the inner cursor. Transactions are processed 100 at a time with respect to data base update. A logical unit of work is defined by a vendor code and invoice number; consequently, a commit is made when either of these variables change oar a termination (successful) of the program

(1) Fetch the following information from AUTO_SCHED (a) based on criterion discussed above:

(a) vendor_code: The vendor code for this expenditure.

(b) inv_nbr: The invoice number referenced by this expenditure.

(2) Fetch the following information from AUTO_SCHED (a) where acpt_flag = 'I', vendor_code = a.vendor_code, and inv_nbr = a.inv_nbr, from step H(1):

(a) oblg_doc_nbr: The obligation document number referenced by this expenditure.

(b) amount: The amount of the expenditure.

(c) scheduled_paymnt_date: The scheduled payment date for the expenditure.

(d) descr: The description for this expenditure.

(3) Retrieve the following information from the OBLIGATION (o) table based on the oblg_doc_nbr in H(2)(a).

(a) item_code: The item code for this obligation.

(b) object_class_code: The object class code for this obligation.

(c) adp_work_code: The COEMIS ADP work code for this obligation.

(d) org_code: The responsible organization code for this obligation.

(e) civ_mil_rvolvg_flag: An indicator of the category of funds for this obligation; civil, military, or revolving fund.

(f) loc_appn_nbr: The appropriation number locally assigned to this obligation.

(g) vendor_code: The vendor code assigned to this obligation.

(h) net_unliq_bal: The net unliquidated balance for this obligation. If an obligation document is not found in the OBLIGATION table, abnormally terminate the program.

(4) Check the obligation document against the following criteria:

(a) Is o.vendor_code = a.vendor_code?

(b) Is o.net_unliq_bal >= a.amount?

If either test fails, update AUTO_SCHED set a.acpt_flag = 'R' for all obligation documents where oblg_doc_nbr = a.oblg_doc_nbr and skip to step 11.

(5) Create the expenditure document for this transaction by forming a concatenated string consisting of:

(a) char 1 - 'F'

(b) char 2-3 - Fiscal month of the system date

(c) char 4-5 - Fiscal year of the system date

(d) char 6-9 - Sequence number for expenditure documents (from RDBMS)

(6) Insert the following record into the EXPENDITURE table:

amount = a.amount

entry_date = sysdate

exp_doc_nbr = document number generated in step 4

init_entry_id = 'U4RFEMHS'

oblg_doc_nbr = a.oblg_doc_nbr

partial_final_flag = 'P'

vendor_addr_code = '01'

vendor_code = a.vendor_code

adp_work_code = o.adp_work_code

status = 'C'

acpt_date = sysdate

del_date = sysdate

descr1 = a.descr

descr2 = null

dssn_nbr = null

inv_nbr = null

pp_nbr = 0

scheduled_paymnt_date = a.scheduled_paymnt_date

tbo_flag = null

(7) Create the interface transaction by inserting the following record into the CMR_4480 table:

acpt_date = sysdate

adp_work_code = o.adp_work_code

amount = a.amount

batch_nbr = '52' for civil, '57' for military, and '47' for revolving fund

civ_mil_rvolvg_flag = o.civ_mil_rvolvg_flag

dest = 'A'

dist_code = '4' (Notes: This is code for WES.)

doc_nbr = expenditure document number generated in step 4.

inter_face_acpt_flag = 'I'

record_code = '02'

trns_code = 'MA' for civil or military and 'MD' for revolving fund.

trns_type_indic = 'E'

item_code = o.item_code

object_class_code = o.object_class_code

org_code = o.org_code

other_ref_nbr = null

pay_coll_code = null for civil and '3' for military or revolving fund.

ref_doc_nbr = a.oblg_doc_nbr for civil or military and null for revolving fund

trf_date = null

trns_month = null

(8) Schedule the expenditures for payment by inserting the following record into the SCHEDULED_PAYMNT table as follows:

adp_work_code = For o.civ_mil_rvolvg_flag = 'C' = first 5 characters of o.adp_work_code concatenated with the string '0000000000'.

For o.civ_mil_rvolvg_flag = 'M' = D level adp work code retrieved from the D_F_FILE table. For 0.civ_mil_rvolvg_flag = 'R' = 'VW0000000000000'

amount = a.amount

exp_doc_nbr = exp_doc_nbr generated i step H(5)

init_entry_id = 'U4RFEMHS'

loc_appn_nbr = o.loc_appn_nbr

object_class_code = o.object_class_code

scheduled_paymnt_date = a.scheduled_paymnt_Date

status = 'A'

vendor_code = a.vendor_code

vendor_addr_code = '01'

contr_mod_nbr = null

contr_nbr = null

trns_date = system date

trnsf_date = null

(9) Set a.acpt_flag = 'A' for all entries in AUTO_SCHED where acpt_flag = 'I' and a.oblg_doc_nbr = current oblg_doc_nbr

(10) Update the OBLIGATION table, set net_unliq_bal = net_unliq_bal - a.amount where o.oblg_doc_nbr = current oblg_doc_nbr

(11) Go to step H(2), repeat until no records are fetched from the cursor.

(12) Search the INVOICE (i) table for a record where i.inv_nbr = a.inv_nbr and i.vendor_code = a.vendor_code.

(a) If the record does not exist, insert the following record into the INVOICE table:

due_date = sysdate

gross_amt = total of all expenditures (a.amount) processed in step H(2) through H(10)

init_entry_id = 'U4RFEMHS'

inv_date = a.inv_nbr

prompt_payment_flag = 'Y'

vendor_code = a.vendor_code

disc_terms_code = null

earnings_period_start_date = null

earnings_period_stop_date = null

rec_date = null

(b) If the record does exist, update the gross_amt adding to it the amount as defined in H(12)(a)(2).

(13) Go to step 1 and repeat until there are no additional combinations of vendor_code and invoice_nbr to be processed by this session.

(14) Print summary report and exit program.

I. **Other Program Notes**:

(1) District code used in step H(7) is specific for the installation (and the data base) V42 processes. This parameter will have to be modified and separate programs maintained if the district codes change.

(2) The expenditure document number generated in step H(5) requires the existence of an ORACLE sequence number named EXPSEQ in the data base.

(3) V42 is designed so that optional performance will be achieved when the number of transactions to be process (i.e., expenditures) per combination of vendor code and invoice number is 99.

8.4 V45_PC

A. **Program Name**: V45

B. **Program Language**: PRO*C

C. **Purpose**: This program reads an input file of obligation document numbers that have an expenditure previously marked as 'final'. The program generates the necessary transactions to liquidate the remaining obligation balance for subsequent processing by COEMIS. The obligation amount is decreased by the balance and the balance is set to 0.

D. **Entry Point**: Executed as a batch program. Command line must contain the userid and password for the data base being processed plus the file name (complete path) for input data (see E).

E. **Input Data Requirements**: Input data to V45 consists of a file containing obligation document numbers that have an expenditure previously flagged as 'final'.

F. **Output**: Program prints the following report:

(1) Listing of the obligation document numbers and amounts for each obligation liquidation.

(2) Total number of transactions processed.

G. **Abnormal Termination**: The program will abnormally terminate upon encountering an ORACLE error. Both a descriptive and the ORACLE error message are printed. Important Note: The program performs only one commit, after processing all transactions; consequently, an error in any transaction will roll back all previously completed transactions.

H. **Program Executions**: The program reads obligation document numbers from the input file in sequential order and performs the following operations:

(1) Retrieve the following information from the OBLIGATION (o) table using the obligation document number (oblg_doc_nbr).

(a) adp_work_code: The COEMIS ADP work code for this obligation.

(b) net_unliq_bal: The net unliquidated balance for this obligation.

(c) civ_mil_rvolvg_flag: An indicator of the category of funds for this obligation, i.e., civil, military, or revolving fund.

(d) object_class_code: The object class code for this obligation.

(e) org_code: The responsible organization code for this obligation.

(f) comt_doc_nbr: The commitment document number referenced by this obligation.

(2) If the obligation document is not found, print an error message citing the document number and skip to the next document on the input file.

(3) Create the interface transaction by inserting the following record into the CMR_4480 table:

acpt_date = sysdate

adp_work_code = o.adp_work_code

amount = o.amount

batch_nbr = '40' for civil and '45' for military

civ_mil_rvolvg_flag = o.civ_mil_rvolvg_flag

dest = 'A'

dist_code = '4' (Note: This is code for WES)

doc_nbr = oblg_doc_nbr from input file

interface_acpt_flag = 'I'

record_code = '02'

trns_code = 'JA'

trns_type_indic = 'O'

item_code = null

object_class_code = o.object_class_code

org_code = o.org_code

other_ref_nbr = null

pay_coll_code = null for civil and '3' for military

ref_doc_nbr = o.comt_doc_nbr

trf_date = null

trns_month = null

(4) Update the OBLIGATION table, set the amount equal to the amount minus the net unliquidated balance and set the net unliquidated balance to 0 for the obligation document

number being processed.

(5) Read another obligation document number from the input file and go to step 1. If an EOF is encountered, commit all transactions and exit from the program.

I. **Other Program Notes**:

(1) District code used in step H(3) is specific for the installation. This parameter will have to be modified for other installations.

(2) This program locks the following tables in exclusive mode during execution: CMR_4480 and OBLIGATION. Care should be taken that VIS45 is run alone to avoid possible deadlock situations.

(3) An attempt to generate this transaction when o.civ_mil_rvolvg_flag = 'R' will result in an error condition.

APPENDIX A - TABLE DEFINITIONS

ACCT_ELM_TYPE

ITEM_CODE	NOT NULL	CHAR(3)
ITEM_DESC	NOT NULL	CHAR(60)
CIVIL		CHAR(1)
CONTRACT		CHAR(1)
IN_HOUSE		CHAR(1)
MIL		CHAR(1)
OTHRF		CHAR(1)
OTHRF_INCOME		CHAR(1)
RVOLVG_81		CHAR(1)

APPROPRIATION

APPN_TITLE	NOT NULL	CHAR(40)
CIV_MIL_RVOLVG_FLAG	NOT NULL	CHAR(1)
LOC_APPN_NBR	NOT NULL	CHAR(9)

AUTO_SCHED

OBLG_DOC_NBR	NOT NULL	CHAR(9)
AMOUNT	NOT NULL	NUMBER(13,2)
VENDOR_CODE	NOT NULL	CHAR(10)
INV_NBR	NOT NULL	CHAR(15)
SCHEDULED_PAYMNT_DATE	NOT NULL	DATE
TRNSF_DATE	NOT NULL	DATE
ACPT_FLAG	NOT NULL	CHAR(1)
DESCR		CHAR(60)

CIVRF

ADP_WORK_CODE	NOT NULL	CHAR(15)
LOC_APPN_NBR	NOT NULL	CHAR(9)

CLASS_OF_OBLIGATION

CIV_CLASS	NOT NULL	CHAR(2)
DESCR	NOT NULL	CHAR(40)
MIL_CLASS	NOT NULL	CHAR(2)
RVOLVG_FUND_CLASS	NOT NULL	CHAR(2)

CMR_4480

ACPT_DATE	NOT NULL	DATE
ADP_WORK_CODE	NOT NULL	CHAR(15)
AMOUNT	NOT NULL	NUMBER(13,2)
BATCH_NBR	NOT NULL	CHAR(2)
CIV_MIL_RVOLVG_FLAG	NOT NULL	CHAR(1)
DEST	NOT NULL	CHAR(1)
DIST_CODE	NOT NULL	CHAR(1)
DOC_NBR	NOT NULL	CHAR(9)
INTERFACE_ACPT_FLAG	NOT NULL	CHAR(1)
RECORD_CODE	NOT NULL	CHAR(2)
TRNS_CODE	NOT NULL	CHAR(2)
TRNS_TYPE_INDIC	NOT NULL	CHAR(1)
ITEM_CODE		CHAR(3)
OBJECT_CLASS_CODE		CHAR(4)
ORG_CODE		CHAR(2)
OTHER_REF_NBR		CHAR(9)
PAY_COLL_CODE		CHAR(1)
REF_DOC_NBR		CHAR(9)
TRF_DATE		DATE
TRNS_MONTH		CHAR(2)
PAY_CODE		CHAR(1)

CONTRACT

AMOUNT	NOT NULL	NUMBER(13,2)
CONTR_MOD_NBR	NOT NULL	NUMBER(3)
CONTR_NBR	NOT NULL	CHAR(16)
CONTR_REF_MOD_NBR	NOT NULL	NUMBER(3)
ENTRY_USERID	NOT NULL	CHAR(12)
VENDOR_CODE	NOT NULL	CHAR(10)
DATE_OF_AWARD		DATE
DISC_TERMS_CODE		CHAR(2)
ENDING_DATE		DATE
FOB_DATE		DATE
RET_PERC		NUMBER(4,2)
RET_PERC_DOC_NBR		CHAR(9)
TAX_ID		CHAR(11)
CONTR_OFFICER_REP		CHAR(25)

DISBURSEMENT

AMOUNT	NOT NULL	NUMBER(13,2)
DISB_DATE	NOT NULL	DATE
DISB_DOC_NBR	NOT NULL	CHAR(9)
EXP_DOC_NBR	NOT NULL	CHAR(9)
CHECK_NBR		CHAR(6)
DOV_NBR		CHAR(6)
TRNSF_DATE		DATE

DISC_TERMS

DISC_DAYS	NOT NULL	NUMBER(2)
DISC_PERC	NOT NULL	NUMBER(4,2)
DISC_TERMS_CODE	NOT NULL	CHAR(2)

D_F_FILE

ADP_WC_D	NOT NULL	CHAR(7)
ADP_WC_F	NOT NULL	CHAR(15)
AMS_CODE		CHAR(11)
ALLOTMENT		CHAR(4)
LOC_APPN_NBR		CHAR(9)
ADP_WC_G	NOT NULL	CHAR(15)

EMPLOYEE

EMPLE_NAME	NOT NULL	CHAR(40)
EMPLE_USERID	NOT NULL	CHAR(12)
ORG_CODE	NOT NULL	CHAR(2)
SSN	NOT NULL	CHAR(9)

EXPENDITURE

AMOUNT	NOT NULL	NUMBER(13,2)
ENTRY_DATE	NOT NULL	DATE
EXP_DOC_NBR	NOT NULL	CHAR(9)
INIT_ENTRY_ID	NOT NULL	CHAR(12)
OBLG_DOC_NBR	NOT NULL	CHAR(9)
PARTIAL_FINAL_FLAG	NOT NULL	CHAR(1)
VENDOR_ADDR_CODE	NOT NULL	CHAR(2)
VENDOR_CODE	NOT NULL	CHAR(10)
ADP_WORK_CODE	NOT NULL	CHAR(15)
STATUS	NOT NULL	CHAR(1)
CYCLE_NBR		CHAR(2)
ACPT_DATE		DATE
DEL_DATE		DATE
DESCR1		CHAR(60)
DESCR2		CHAR(60)
DISC_TERMS_CODE		CHAR(2)
DSSN_NBR		CHAR(4)
INV_NBR		CHAR(15)
PARTIAL_PAY_NBR		NUMBER(3)
SCHEDULED_PAYMNT_DATE		DATE
TBO_FLAG		CHAR(1)

INVOICE

DUE_DATE	NOT NULL	DATE
GROSS_AMT	NOT NULL	NUMBER(13,2)
INIT_ENTRY_ID	NOT NULL	CHAR(12)
INV_DATE	NOT NULL	DATE
INV_NBR	NOT NULL	CHAR(15)
PROMPT_PAYMNT_FLAG	NOT NULL	CHAR(1)
VENDOR_CODE	NOT NULL	CHAR(10)
DISC_TERMS_CODE		CHAR(2)
EARNINGS_PERIOD_START_DATE		DATE
EARNINGS_PERIOD_STOP_DATE		DATE
REC_DATE		DATE

OBJECT_CLASS_CODES

DESCR	NOT NULL	CHAR(40)
OBJECT_CLASS_CODE	NOT NULL	CHAR(4)
CIV		CHAR(1)
MIL		CHAR(1)
RVOLVG_FUND		CHAR(1)

OBLIGATION

AMOUNT	NOT NULL	NUMBER(13,2)
ENTRY_DATE	NOT NULL	DATE
INIT_ENTRY_ID	NOT NULL	CHAR(12)
ITEM_CODE	NOT NULL	CHAR(3)
OBJECT_CLASS_CODE	NOT NULL	CHAR(4)
OBLG_DOC_NBR	NOT NULL	CHAR(9)
VENDOR_ADDR_CODE	NOT NULL	CHAR(3)
VENDOR_CODE	NOT NULL	CHAR(10)
NET_UNLIQ_BAL	NOT NULL	NUMBER(13,2)
ADP_WORK_CODE	NOT NULL	CHAR(15)
COMT_DOC_NBR	NOT NULL	CHAR(9)
ORG_CODE	NOT NULL	CHAR(2)
CIV_MIL_RVOLVG_FLAG	NOT NULL	CHAR(1)
STATUS	NOT NULL	CHAR(1)
LOC_APPN_NBR	NOT NULL	CHAR(9)
CONTR_MOD_NBR		NUMBER(3)
CONTR_NBR		CHAR(16)

ORGANIZATION

ORG_CODE	NOT NULL	CHAR(2)
ORG_INDIC	NOT NULL	CHAR(1)
ORG_NAME	NOT NULL	CHAR(40)
PARENT_CODE		CHAR(2)

RECURRING_EXP

AMOUNT	NOT NULL	NUMBER(13,2)
CONTR_MOD_NBR	NOT NULL	NUMBER(3)
CONTR_NBR	NOT NULL	CHAR(16)
ENTRY_DATE	NOT NULL	DATE
EXPR_DATE	NOT NULL	DATE
INIT_ENTRY_ID	NOT NULL	CHAR(12)
ITEM_CODE	NOT NULL	CHAR(3)
NO_PAYMNTS	NOT NULL	NUMBER(2)
OBJECT_CLASS_CODE	NOT NULL	CHAR(4)
OBLG_DOC_NBR	NOT NULL	CHAR(9)
PAYMNT_PERIOD	NOT NULL	NUMBER(2)
START_DATE	NOT NULL	DATE
STATUS	NOT NULL	CHAR(1)
VENDOR_ADDR_CODE	NOT NULL	CHAR(2)
VENDOR_CODE	NOT NULL	CHAR(10)
ADP_WORK_CODE	NOT NULL	CHAR(15)
DESCR1		CHAR(60)
DESCR2		CHAR(60)
NEXT_EXP_DATE		DATE

ROLE_FUNCT

ROLE	NOT NULL	CHAR(12)
FUNCTCODE	NOT NULL	CHAR(9)

SCHEDULED_PAYMNT

ADP_WORK_CODE	NOT NULL	CHAR(15)
AMOUNT	NOT NULL	NUMBER(13,2)
EXP_DOC_NBR	NOT NULL	CHAR(9)
INIT_ENTRY_ID	NOT NULL	CHAR(12)
LOC_APPN_NBR	NOT NULL	CHAR(9)
OBJECT_CLASS_CODE	NOT NULL	CHAR(4)
SCHEDULED_PAYMNT_DATE	NOT NULL	DATE
STATUS	NOT NULL	CHAR(1)
VENDOR_ADDR_CODE	NOT NULL	CHAR(2)
VENDOR_CODE	NOT NULL	CHAR(10)
CONTR_MOD_NBR		NUMBER(3)
CONTR_NBR		CHAR(16)
TRNS_DATE		DATE
TRNSF_DATE		DATE

TRAVEL_EXP

ACPT_FLAG	NOT NULL	CHAR(1)
ADV_AMT	NOT NULL	NUMBER(13,2)
EXP_AMT	NOT NULL	NUMBER(13,2)
OBLG_DOC_NBR	NOT NULL	CHAR(9)
VENDOR_CODE	NOT NULL	CHAR(10)
TRNSF_DATE		DATE

USERFUNCT

FUNCTCODE	NOT NULL	CHAR(9)
FUNCTDESC		CHAR(50)
FUNCTYPE		CHAR(1)
MNUORD		CHAR(2)
PARCODE		CHAR(9)

VENDOR

VENDOR_CODE	NOT NULL	CHAR(10)
VENDOR_NAME	NOT NULL	CHAR(40)
VENDOR_TAX_ID		CHAR(11)
VENDOR_TYPE		CHAR(1)

VENDOR_ADDR

VENDOR_ADDR_CODE	NOT NULL	CHAR(2)
VENDOR_CODE	NOT NULL	CHAR(10)
VENDOR_ADDR1		CHAR(40)
VENDOR_ADDR2		CHAR(40)
VENDOR_CITY		CHAR(20)
VENDOR_COUNTRY		CHAR(2)
VENDOR_PHONE		CHAR(10)
VENDOR_STATE		CHAR(2)
VENDOR_ZIP		CHAR(10)

VROLE

KEYCODE	NOT NULL	CHAR(1)
ROLE	NOT NULL	CHAR(12)
DESCRIPTION		CHAR(40)
PASSWORD		CHAR(20)
LAST_UPDATE		DATE

VUSERID_ROLE

ROLE	NOT NULL	CHAR(12)
USERID	NOT NULL	CHAR(12)

APPENDIX B - DATA DICTIONARY

ACCT_ELM_TYPE

Static table which contains all valid accounting element types (item codes).

ITEM_CODE

A three digit numeric code used to identify a specific item or expense, or type of cost charged to an account in COEMIS. Same as the COEMIS accounting element.

ITEM_DESC

A brief description of the item code.

CIVIL

Denotes the item code is civil.

CONTRACT

Denotes the item code is contract.

IN_HOUSE

Denotes the item code is in house.

MIL

Denotes the item code is military.

OTHRF

Denotes the item code is other revolving fund.

OTHRF_INCOME

Denotes the item code is other revolving fund income.

RVOLVG_81

Denotes the item code is revolving fund 81.

APPROPRIATION

Static table which contains all valid appropriation numbers for Civil, Military, and Revolving Fund.

APPN_TITLE

A brief description of the appropriation.

CIV_MIL_RVOLVG_FLAG

A code which displays the type of appropriation. C is for Civil; M is for Military; and R is for Revolving Fund.

LOC_APPN_NBR

A nine character code which identifies the appropriation from which the work is funded.

AUTO_SCHED

Contains obligation and expenditure information for automatically scheduling CitiCorp travel.

OBLG_DOC_NBR

A unique number that is assigned to the obligation transaction when it is created. The obligation number contains nine characters and the first is always an "E". The second and third positions identify the class of obligation. The fourth and fifth positions identify the fiscal year the obligation is created. The last four positions are an alphanumeric sequence number.

AMOUNT

The amount of the obligation.

VENDOR_CODE

Code that represents the WES identifier for a specific vendor and it is unique to the vendor.

INV_NBR

A vendor assigned number to designate a unique invoice for a particular vendor.

SCHEDULED_PAYMNT_DATE

The date the expenditure is scheduled to be paid.

TRNSF_DATE

The date the transactions are uploaded to the vendor system data base.

ACPT_FLAG

A code which denotes whether or not the transaction is in progress, accepted, or rejected. "I" represents in progress, "A" represents accepted, and "R" represents rejected.

DESCR

The description associated with the transaction. May include the traveler's name.

CIVRF

Contains all 15 digit adp work codes for Civil and Revolving Fund jobs.

ADP_WORK_CODE

A fifteen digit alphanumeric code representing an accounting classification (account or job number). It identifies the type of funding, project, or account.

LOC_APPN_NBR

A nine character code which identifies the appropriation from which the work is funded.

CLASS_OF_OBLIGATION

Static table which contains all valid class of obligations for Civil, Military, and Revolving Fund.

CIV_CLASS

Class of obligation code for Civil.

DESCR

Brief description for the obligation class.

MIL_CLASS

Class of obligation code for Military.

RVOLVG_FUND_CLASS

Class of obligation code for Revolving Fund.

CMR_4480

Contains obligation, expenditure, and disbursement transactions in 4480 format for updating COEMIS.

ACPT_DATE

Date transaction is entered into the system.

ADP_WORK_CODE

A fifteen digit alphanumeric code representing an accounting classification (account or job number). It identifies the type of funding, project, or account.

AMOUNT

The transaction amount sent to COEMIS for a particular type of document.

BATCH_NBR

A two character numeric code which identifies a batch in COEMIS.

CIV_MIL_RVOLVG_FLAG

A code which displays the type of appropriation. C is for Civil; M is for Military; and R is for Revolving Fund.

DEST

Determines if the transaction goes to COEMIS or Funds Control. "A" represents COEMIS and "F" represents Funds Control.

DIST_CODE

A one character numeric code to designate a Corps of Engineers (COE) facility. The district code for WES is 4.

DOC_NBR

A nine digit alphanumeric code which represents different type of documents. A "R" in the first position of the nine digit number signifies a Commitment Document Number. An "E" refers to an Obligation Document Number. A "F" refers to an Expenditure Document Number. A "K" refers to a Disbursement Document Number.

INTERFACE_ACPT_FLAG

A flag which is set to 'I' if the transaction has not been sent to COEMIS or 'A' if the transaction has been sent to COEMIS.

RECORD_CODE

A two character numeric code assigned to an input document which indicates the format of the record for processing data.

TRNS_CODE

A two character alpha code which designates the type of action to be performed on a transaction.

TRNS_TYPE_INDIC

Identifies the document type.

ITEM_CODE

A three digit numeric code used to identify a specific item or expense, or type of cost charged to an account in COEMIS. Same as the COEMIS accounting element.

OBJECT_CLASS_CODE

A four digit numeric code which represents object classification. It is a method to provide for the classification of obligations or expenditures representing the different types of services, goods, or other items being procured or consumed.

ORG_CODE

A two digit alphanumeric code assigned to a particular organizational element of a Corps facility.

OTHER_REF_NBR

A nine digit alphanumeric code which represents different types of documents and references the document number.

PAY_COLL_CODE

A one digit numeric code used to designate a type of transaction payment.

REF_DOC_NBR

A nine digit alphanumeric code which represents different types of documents and references the document number.

TRF_DATE

The day the transaction is sent to COEMIS.

TRNS_MONTH

The calendar month the transaction is sent to COEMIS.

PAY_CODE

Used to designate transactions that are "within or outside the government" for object class purposes.

CONTRACT

Contains detail information for a specific contract.

AMOUNT

The gross amount of the contract.

CONTR_MOD_NBR

A sequential number that is entered each time a modification is made to the contract.

CONTR_NBR

A unique number assigned to a legally enforceable binding agreement between two or more parties for the supply of certain goods or services.

CONTR_REF_MOD_NBR

The "parent" or first contract mod number for a contract that is within the same scope of work.

ENTRY_USERID

A userid that is associated with the employee name who entered the contract information into the system.

VENDOR_CODE

A 10 digit code that identifies products/services provided to the Corp of Engineers.

DATE_OF_AWARD

Date the contract is actually awarded.

DISC_TERMS_CODE

Each contract as negotiated with the contractor can have various terms offered by the government and contractor. Discount terms allow the government to pay less than the billed earnings when they are paid early or within a specified time period. These terms may also be used in the invoice. These codes are assigned to identify the combination of negotiated terms and discount percentage.

ENDING_DATE

Date the contract is expected to end.

FOB_DATE

Date the goods were placed or delivered to the FOB destination.

RET_PERC

The percentage of a contractor's earnings, agreed upon during contract negotiation, that will be withheld until final delivery.

RET_PERC_DOC_NBR

Document number assigned to the retained earnings withheld and not yet disbursed to the contractor.

TAX_ID

Federal tax id number that is assigned to a corporation or company. It is used to report earnings to the IRS.

CONTR_OFFICER_REP

The government employee assigned to monitor a contract and to act as an agent of the contracting officer.

DISBURSEMENT

Contains detail disbursement information for an expenditure.

AMOUNT

The amount to be disbursed to the vendor.

DISB_DATE

The date the document is disbursed.

DISB_DOC_NBR

The document number assigned to the COEMIS transaction to record the liquidation of the expenditure and the disbursement of the cash to the vendor.

EXP_DOC_NBR

A number used to identify the cost or expenditure transaction created when goods or services are received.

CHECK_NBR

A unique number assigned to each check being disbursed.

DOV_NBR

The disbursing officer's voucher number assigned to the batch of transactions being disbursed.

TRNSF_DATE

The date the disbursement transaction is transferred from the disbursing system to COEMIS.

DISC_TERMS

Static table which contains all available discount terms possible for a contract or invoice.

DISC_DAYS

The number of days used to calculate the valid discount.

DISC_PERC

The actual percentage of the discount.

DISC_TERMS_CODE

Each contract as negotiated with the contractor can have various terms offered by the government and contractor. Discount terms allow the government to pay less than the billed earnings when they are paid early or within a specified time period. These terms may also be used in the invoice. These codes are assigned to identify the combination of negotiated terms and discount percentage.

D_F_FILE

Contains all D, F, and G level adp work codes for Military jobs.

ADP_WC_D

Military "D" level ADP Work Code.

ADP_WC_F

Military "F" level ADP Work Code.

AMS_CODE

Army Management System code assigned to a funding document or program.

ALLOTMENT

The administrative allocation of the congressionally approved appropriated funds.

LOC_APPN_NBR

A nine character code which identifies the appropriation from which the work is funded.

ADP_WC_G

Military "G" level ADP Work Code.

EMPLOYEE

Contains all employees having access to the Vendor Information System.

EMPLE_NAME

Name of WES employee.

EMPLE_USERID

ID associated with the employee name.

ORG_CODE

A two digit alphanumeric code assigned to a particular organizational element of a Corps facility.

SSN

The social security number that is associated with the employee.

EXPENDITURE

Contains detail information for an expenditure.

AMOUNT

Amount recorded as cost/expenditure for goods and services placed or consumed.

ENTRY_DATE

Date the expenditure is entered into the system.

EXP_DOC_NBR

A unique number used to identify a transaction record made by the receipt of goods or services. The expenditure number contains nine characters and the first character is always an 'F'.

INIT_ENTRY_ID

The userid of the employee who entered the expenditure into the system.

OBLG_DOC_NBR

A unique number that is assigned to the obligation transaction when it is created. The obligation number contains nine characters and the first character is always an 'E'. This particular obligation number is the number that is being expended against.

PARTIAL_FINAL_FLAG

A flag which denotes whether the payment is partial or final. It is marked "F" for final or "P" for partial.

VENDOR_ADDR_CODE

A number uniquely identifying a particular address for a vendor.

VENDOR_CODE

Code that represents the WES identifier for a specific vendor and it is unique to the vendor.

ADP_WORK_CODE

A fifteen digit alphanumeric code representing an accounting classification (account or job number). It identifies the type of funding, project, or account.

STATUS

This flag denotes that the expenditure is either certified or canceled. A "C" represents certified and a "X" represents canceled.

CYCLE_NBR

COEMIS required number that equates to the run number for an update.

ACPT_DATE

The date the goods and services are accepted as represented by the receiving report and expenditure documents.

DEL_DATE

The date the goods are to be delivered.

DESCR1

Line 1 of a brief description that is entered by the user.

DESCR2

Line 2 of a brief description that is entered by the user.

DISC_TERMS_CODE

Each contract as negotiated with the contractor can have various terms offered by the government and the contractor. Discount terms allow the government to pay less than the billed earnings when they are paid early or within a specified time period. These terms may also be used in the invoice. These codes are assigned to identify the combination of negotiated terms and discount percentage.

DSSN_NBR

Disbursing station number assigned to the Department of Treasury.

INV_NBR

A vendor assigned number to designate a unique invoice for a particular vendor.

PARTIAL_PAY_NBR

A RMO number assigned to the consecutive payment of a contract that has progressive or periodic earnings. Each earnings period or earnings request will be assigned a partial payment number for their control of the sequence of the disbursements.

SCHEDULED_PAYMNT_DATE

The date the expenditure is scheduled to be paid.

TBO_FLAG

A flag that tells the disbursing system that the expenditure is a "transaction by

others" and creates a unique accounting entry for the recording of the disbursement made by another government office for WES.

INVOICE

Contains detail invoice information for a particular vendor.

DUE_DATE

The date the invoice is due.

GROSS_AMT

The total amount for the invoice.

INIT_ENTRY_ID

The userid of the employee who entered the invoice information into the system.

INV_DATE

The date of the invoice.

INV_NBR

A vendor assigned number to designate an invoice from a particular vendor.

PROMPT_PAYMNT_FLAG

A flag which denotes whether or not the invoice is subject to prompt payment. A "Y" represents yes and a "N" represents no.

VENDOR_CODE

Code that represents the WES identifier for a specific vendor and it is unique to the vendor.

DISC_TERMS_CODE

Each contract as negotiated with the contractor can have various terms offered by the government and contractor. Discount terms allow the government to pay less than the billed earnings when they are paid early or within a specified time period. These codes are assigned to identify the combination of negotiated terms and discount percentage.

EARNINGS_PERIOD_START_DATE

The date the contractor's earnings start.

EARNINGS_PERIOD_STOP_DATE

The ending date the contractor's invoice is requesting payment through.

REC_DATE
The date the goods are received.

OBJECT_CLASS_CODES
Static table which contains all valid object class codes.

DESCR
A brief description of the object class code.

OBJECT_CLASS_CODE
A four digit numeric code which represents object classification. It is a method to provide for the classification of obligations or expenditures representing the different types of services, goods, or other items being procured or consumed.

CIV
Denotes a civil object class code.

MIL
Denotes a military obje class code.

RVOLVG_FUND
Denotes a revolving fund object class code.

OBLIGATION
Contains detail information for an obligation.

AMOUNT
The amount of the obligation.

ENTRY_DATE
The date the obligation is entered into the system.

INIT_ENTRY_ID
The userid of the employee who entered the obligation into the system.

ITEM_CODE
A three digit numeric code used to identify a specific item or expense, or type of cost charged to an account in COEMIS.

OBJECT_CLASS_CODE
A four digit numeric code which represents object classification. It is a method to provide for the classification of obligations or expenditures representing the

different types of services, goods, or other items being procured or consumed.

OBLG_DOC_NBR

A unique number that is assigned to the obligation transaction when it is created. The obligation number contains nine characters and the first is always an "E". The second and third positions identify the class of obligation. The fourth and fifth positions identify the fiscal year the obligation is created. The last four positions are an alphanumeric sequence number.

VENDOR_ADDR_CODE

A number uniquely identifying a particular address for a vendor.

VENDOR_CODE

Code that represents the WES identifier for a specific vendor and it is unique to the vendor.

NET_UNLIQ_BAL

This is the remaining balance left in the obligation. If any expenditures are created against the obligation, the expenditure amount is subtracted from the net unliquidated balance.

ADP_WORK_CODE

A fifteen digit alphanumeric code representing an accounting classification (account or job number). It identifies the type of funding, project, or account.

COMT_DOC_NBR

A nine digit alphanumeric number which the obligation references as a reference document number. Obligations will liquidate the commitment document's net unliquidated balance. This number beings with an "R".

ORG_CODE

A two digit alphanumeric code assigned to a particular organizational element of a Corps facility.

CIV_MIL_RVOLVG_FLAG

A code which displays the type of appropriation. C is for civil; M for military; and r is for revolving.

STATUS

A code which denotes whether the obligation is certified or canceled. A "C" represents certified and a "X" represents canceled.

LOC_APPN_NBR

A nine digit alphanumeric code which identifies the appropriation from which the work is funded.

CONTR_MOD_NBR

A sequential number that is entered each time a modification is made to the contract.

CONTR_NBR

A unique number which represents a legally enforceable binding agreement between two or more parties for the supply of certain goods or services.

ORGANIZATION

Static table which contains all valid organization codes for WES.

ORG_CODE

A two digit alphanumeric code assigned to a particular organizational element of a Corps activity.

ORG_INDIC

Identifies the organization's work unit.

ORG_NAME

Name of the organization.

PARENT_CODE

The organization code structure is designed in a way that every child org code must belong to a higher level org code or a parent org code. A parent org code can have many different children org codes or just one.

RECURRING_EXP

Contains detail information for a recurring expenditure.

AMOUNT

The expenditure amount to be used when the recurring expenditure is created.

CONTR_MOD_NBR

A sequential number that is entered each time a modification is made to a contract.

CONTR_NBR

A number which represents a legally enforceable binding agreement between two or more parties for the supply of certain goods or services.

ENTRY_DATE

The date the recurring expenditure is entered into the system.

EXPR_DATE

The date the expenditure is created.

INIT_ENTRY_ID

The userid of the employee who entered the recurring expenditure into the system.

ITEM_CODE

A three digit numeric code used to identify a specific item or expense, or type of cost charged to an account in COEMIS.

NO_PAYMNTS

A numeric number which denotes the number of payments that are made for the recurring expenditure.

OBJECT_CLASS_CODE

A four digit numeric code which represents object classification. It is a method to provide for the classification of obligations or expenditures representing the different types of services, goods, or other items being procured or consumed.

OBLG_DOC_NBR

A unique number that is assigned to the obligation transaction when it is created. The number contains nine characters and the first is always an "E". The obligation number that the recurring expenditure references and liquidates.

PAYMNT_PERIOD

The two digit numeric code that represents how often an expenditure is to be generated for payment.

START_DATE

Date the payments are to start.

STATUS

A one character code which represents whether the recurring expenditure is active or inactive. An "A" denotes active and an "I" denotes inactive.

VENDOR_ADDR_CODE

A number uniquely identifying a particular address for a vendor.

VENDOR_CODE

Code that represents the WES identifier for a specific vendor and it is unique to the vendor.

ADP_WORK_CODE

A fifteen digit alphanumeric code representing an accounting classification (account or job number). It identifies the type of funding, project, or account.

DESCR1

Line 1 of a brief description that the user enters.

DESCR2

Line 2 of a brief description that the user enters.

NEXT_EXP_DATE

A date generated by the system which determines the next date the expenditure is created.

ROLE_FUNCT

Static table which contains all functional roles and corresponding screen codes.

ROLE

The functional role the system allows users to play and thus determines the appropriate screens the user may access.

FUNCTCODE

List of screens each role may access.

SCHEDULED_PAYMNT

Contains detail information for expenditures scheduled for payment.

ADP_WORK_CODE

A fifteen digit alphanumeric code representing an accounting classification (account or job number). It identifies the type of funding, project, or account.

AMOUNT

The amount the expenditure is scheduled for payment.

EXP_DOC_NBR

A unique number used to identify a transaction record made by the receipt of

goods or services. This number is 9 characters and always begins with an "F". The expenditure number is the reference document number used by the disbursing transaction.

INIT_ENTRY_ID

The userid of the employee who scheduled the expenditure for payment.

LOC_APPN_NBR

A nine digit alphanumeric code which identifies the appropriation from which the work is funded.

OBJECT_CLASS_CODE

A four digit numeric code which represents object classification. It is a method to provide for the classification of obligations or expenditures representing the different types of services, goods, or other items being procured or consumed.

SCHEDULED_PAYMNT_DATE

The date the expenditure is scheduled to be paid.

STATUS

A one character code which represents whether the scheduled payment is canceled, inactive, or active. 'C' represents canceled, 'I' represents inactive , and 'A' represents active.

VENDOR_ADDR_CODE

A number uniquely identifying a particular address for a vendor.

VENDOR_CODE

Code that represents the WES identifier for a specific vendor and it is unique to the vendor.

CONTR_MOD_NBR

A sequential number that is entered each time a modification is made to a contract.

CONTR_NBR

A unique number which represents a legally enforceable binding agreement between two or more parties for the supply of certain goods or services.

TRNS_DATE

The date the transaction is created.

TRNSF_DATE

The date the transaction is sent to COEMIS.

TRAVEL_EXP

Contains obligation and expenditure information for generating travel expenditures.

ACPT_FLAG

A code which denotes whether or not the travel expenditure is in progress, accepted or rejected. "I" represents in progress, "A" represents accepted, and "R" represents rejected.

ADV_AMT

The amount of the travel advance.

EXP_AMT

The amount of the expenditure.

OBLG_DOC_NBR

A unique number that is assigned to the obligation transaction when it is created. The number contains nine characters and the first is always an "E". The obligation number that the travel expenditure is against or references as the reference document number.

VENDOR_CODE

Code that represents the WES identifier for a specific vendor and it is unique to the vendor. For travel it is the employee's social security number.

TRNSF_DATE

The date the transactions are uploaded to the vendor system data base.

USERFUNCT

Static table which contains menuing information.

FUNCTCODE

Name of the SQL*FORM accessed in menu or a menu screen.

FUNCTDESC

Description of the SQL*FORM.

FUNCTYPE

Determines if the functcode is a menu or an SQL*FORM. 'F' denotes form and 'M' denotes menu.

MNUORD
Determines the hierarchy of the menu.

PARCODE
Determines where the SQL*FORM is located in the menu hierarchy.

VENDOR
Contains a list of all the available vendor codes.

VENDOR_CODE
Code that represents the WES identifier for a specific vendor and it is unique to the vendor.

VENDOR_NAME
The vendor name that is associated with the vendor code.

VENDOR_TAX_ID
Federal tax id number that is assigned to the vendor. It is used to report earnings to the IRS.

VENDOR_TYPE
This denotes whether the vendor is commercial or noncommercial. "C" represents commercial and "N" represents noncommercial.

VENDOR_ADDR
Contains addresses for each vendor.

VENDOR_ADDR_CODE
A number uniquely identifying a particular address for a vendor.

VENDOR_CODE
Code that represents the WES identifier for a specific vendor and it is unique to the vendor.

VENDOR_ADDR1
Line 1 of the vendor's street address.

VENDOR_ADDR2
Line 2 of the vendor's street address.

VENDOR_CITY
The city the vendor is located.

VENDOR_COUNTRY
The country the vendor is located.

VENDOR_PHONE
The area code and phone number where the vendor can be reached.

VENDOR_STATE
The state the vendor is located.

VENDOR_ZIP
The zip the vendor is located.

VROLE

Static table which contains all available roles and their descriptions.

KEYCODE
The first letter of available roles in VIS.

ROLE
List of roles available in VIS.

DESCRIPTION
A brief description of each role.

PASSWORD
Password encryption used to connect and disconnect roles to VIS.

LAST_UPDATE
Last time this table is updated.

VUSERID_ROLE

Static table which contains userids and the roles they have access to.

ROLE
Roles each userid may access.

USERID
The userid of the employee.

Waterways Experiment Station Cataloging-In-Publication Data

Duett, Patti S.
Vendor Information System (VIS) systems manual / by Patti S. Duett, Monique F. Harrison.
156 p. : ill. ; 28 cm. -- (Technical report ; ITL-92-5)
1. Vendor Information System (Information retrieval system) 2. SQL (Computer program language) 3. Management information systems. 4. Programming languages (Electronic computers) I. Harrison, Monique F. II. U.S. Army Engineer Waterways Experiment Station. III. Title. IV. Series: Technical report (U.S. Army Engineer Waterways Experiment Station) ; ITL-92-5.
TA7 W34 no.ITL-92-5